TECHNISCHE UNIVERSITÄT
CHEMNITZ

Diplomarbeit

Semantic Web – basiertes System zur Unterstützung von Workflow- und Projektmanagement

Vorgelegt von André Langer

30. November 2007

Fakultät für Informatik
Professur Verteilte und Selbstorganisierende Rechnersysteme
Prof. Dr. Martin Gaedke

Betreut durch:

Prof. Dr. Martin Gaedke
Dr. Jörg Anders

Eidesstattliche Erklärung

Hiermit erkläre ich an Eides Statt, dass ich die vorliegende Arbeit selbstständig angefertigt, nicht anderweitig zu Prüfungszwecken vorgelegt und keine anderen als die angegebenen Hilfsmittel verwendet habe. Sämtliche wissentlich verwendeten Textausschnitte, Zitate oder Inhalte anderer Verfasser wurden ausdrücklich als solche gekennzeichnet.

Chemnitz, den 30. November 2007

Zusammenfassung

Mit mehr als 120 Millionen registrierten Internetadressen (Stand: März 2007) symbolisiert das Internet heutzutage das größte Informationsmedium unserer Zeit. Täglich wächst das Internet um eine unüberschaubare Menge an Informationen. Diese Informationen sind häufig in Dokumenten hinterlegt, welche zur Auszeichnung die Hypertext Markup Language verwenden. Seit Beginn der Neunziger Jahre hat sich dieses System bewährt, da dadurch der einzelne Nutzer in die Lage versetzt wird, auf einfache und effiziente Weise Dokumentinhalte mit Darstellungsanweisungen zu versehen und diese eigenständig im Internet zu veröffentlichen. Diese Layoutinformationen können bei Abruf der entsprechenden Ressource durch ein Computerprogramm leicht ausgewertet und zur Darstellung der Inhalte genutzt werden. Obwohl sowohl die Layoutinformationen als auch die eigentlichen Dokumentinhalte in einem textuellen Format vorliegen, konnten die Nutzertextinhalte durch eine Maschine bisher nur sehr eingeschränkt verarbeitet werden. Während es menschlichen Nutzern keinerlei Probleme bereitet, die Bedeutung einzelner Texte auf einer Webseite zu identifizieren, stellen diese für einen Rechner prinzipiell nur eine Aneinanderreihung von ASCII-Zeichen dar.

Sobald es möglich werden würde, die Bedeutung von Informationen durch ein Computerprogramm effizient zu erfassen und weiterzuverarbeiten, wären völlig neue Anwendungen mit qualitativ hochwertigeren Ergebnissen im weltweiten Datennetz möglich. Nutzer könnten Anfragen an spezielle Agenten stellen, welche sich selbstständig auf die Suche nach passenden Resultaten begeben; Informationen verschiedener Informationsquellen könnten nicht nur auf semantischer Ebene verknüpft, sondern daraus sogar neue, nicht explizit enthaltene Informationen abgeleitet werden. Ansätze dazu, wie Dokumente mit semantischen Metadaten versehen werden können, gibt es bereits seit einiger Zeit. Lange umfasste dies jedoch die redundante Bereitstellung der Informationen in einem eigenen Dokumentenformat, weswegen sich keines der Konzepte bis in den Privatbereich durchsetzen konnte und als Endkonsequenz in den vergangenen Monaten besonderes Forschungsinteresse darin aufkam, Möglichkeiten zu finden, wie semantische Informationen ohne großen Zusatzaufwand direkt in bestehende HTML-Dokumente eingebettet werden können.

Die vorliegende Diplomarbeit möchte diese neuen Möglichkeiten im Bereich des kollaborativen Arbeitens näher untersuchen. Ziel ist es dazu, eine Webapplikation zur Abwicklung typischer Projektmanagement-Aufgaben zu entwickeln, welche jegliche Informationen unter einem semantischen Gesichtspunkt analysieren, aufbereiten und weiterverarbeiten kann und unabhängig von der konkreten Anwendungsdomain und Plattform systemübergreifend eingesetzt werden kann. Die Konzepte Microformats und RDFa werden dabei besonders herausgestellt und nach Schwächen und zukünftigen Potentialen hin untersucht.

Abstract

The World Wide Web supposably symbolizes with currently more than 120 million registered internet domains (March 2007) the most comprehensive information reference of all times. The amount of information available increases by a storming bulk of data ever day. Those information is often embedded in documents which utilize the Hypertext Markup Language. This enables the user to mark out certain layout properties of a text in an easy and efficient fashion and to publish the final document containing both layout and data information. A computer application is then able to extract style information from the document resource and to use it in order to render the resulting website. Although layout information and data are both equally represented in a textual manner, a machine was hardly capable of processing user content so far. Whereas human consumers have no problem to identify and understand the sense of several paragraphs on a website, they basically represent only a concatenation of ASCII characters for a machine.

If it were possible to efficiently disclose the sense of a word or phrase to a computer program in order to process it, new astounding applications with output results of high quality would be possible. Users could create queries for specialized agents which autonomously start to search the web for adequate result matches. Moreover, the data of multiple information sources could be linked and processed together on a semantic level so that above all new, not explicitly stated information could be inferred. Approaches already exist, how documents could be enhanced with semantic meta-data, however, many of these involve the redundant provision of those information in a specialized document format. As a consequence none of these concepts succeeded in becoming a widely used method and research started again to find possibilities how to embed semantic annotations without huge additional efforts in an ordinary HTML document.

The present thesis focuses on an analysis of these new concepts and possibilities in the area of collaborative work. The objective is to develop the prototype of a web application with which it is possible to manage typical challenges in the realm of project and workflow management. Any information available should be processable under a semantic viewpoint which includes analysis, conditioning and reuse independently from a specific application domain and a certain system platform. Microformats and RDFa are two of those relatively new concepts which enable an application to extract semantic information from a document resource and are therefore particularly exposed and compared with respect to advantages and disadvantages in the context of a “Semantic Web”.

Danksagung

"Whatever you do will be unimportant, but it is very important that you do it." ~ Mahatma Gandhi

An der vorliegenden Diplomarbeit haben viele Personen einen direkten oder indirekten Einfluss gehabt, denen ich diesen kurzen Abschnitt widmen möchte. Die Zeit von sechs Monaten zur Anfertigung einer Diplomarbeit scheint zu Beginn lang und das gewählte Thema scheint das einzig Wichtige zu sein, was den Inhalt der Arbeit bestimmt. Ich persönlich habe während der Anfertigung der Arbeit Erfahrungen gemacht, die über diesen inhaltlichen Fokus weit hinausgehen. Sicherlich stand zu Beginn die Frage im Raum, mit welchem Thema ich mich in den kommenden Monaten beschäftigen möchte, und ich bin froh dieses sehr zukunftsträchtige Thema gewählt zu haben, da ich mich sonst sicherlich nicht derart intensiv damit auseinander gesetzt hätte. Viel interessanter war für mich jedoch zu sehen, wie ich mich persönlich weiterentwickelt habe. Die Diplomarbeit war dabei ein dynamischer Prozess, der durch Erfolge aber auch durch kleine und große Probleme geprägt war, die zu Beginn nicht immer abzusehen, aber letztendlich alle überwunden werden konnten. Ich habe im Zuge dessen neue, interessante Methoden, Ansätze und Kontakte kennen gelernt, die mir im späteren Berufsleben von großem Nutzen sein können.

An erster Stelle möchte ich mich dazu bei meinem Betreuer Prof. Dr. Martin Gaedke bedanken, der wesentlichen Anteil an der gemeinsamen Entwicklung des Diplomarbeitsthemas hatte und mehrfach seinen Feierabend entfallen ließ, um mit mir stundenlang aktuelle Ergebnisse bis in die Nacht zu diskutieren. Außerdem danke ich dem gesamten Forschungsteam der Professur Verteilte und Selbstorganisierende Rechnersysteme der Technischen Universität Chemnitz, welche sich in der zurückliegenden Zeit bemerkenswert einsetzten, um mit neuen Lehrmethoden alle Studierende dazu zu ermutigen, über den Tellerrand hinauszuschauen und die zugrunde liegenden Zusammenhänge zu begreifen, um diese in der Praxis später eigenständig anwenden zu können.

Mein weiterer Dank gilt meinen Großeltern, welche mich in der Diplomzeit mit allen nur erdenklichen Hilfen unterstützten, damit ich mein Studium mit einem guten Ergebnis abschließen kann. Ebenso möchte ich mich bei meinen Freunden bedanken, die mich immer wieder ermutigt und motiviert haben, mein Ziel zu verfolgen. Insbesondere seien hier Maja Heidrich, bei der ich mich für das Korrekturlesen der Diplomarbeit bedanken möchte, sowie Thomas Fichtner genannt, der mir immer wieder Feedback bei der Implementierung des Prototypen gab.

Mein abschließender Dank gilt dem Unternehmen der 09111 Studio Chemnitz GmbH&Co. KG, welche auch in den letzten zwei Monaten eine sehr flexible Arbeitszeitgestaltung ermöglichten, um in beiderseitigem Interesse das Studium erfolgreich abschließen zu können, sowie der Stiftung der Deutschen Wirtschaft, bei der ich mich für die Förderung im Rahmen des Stipendiatenprogramms in den zurückliegenden vier Jahren und das entgegengebrachte Vertrauen bedanken möchte.

Die Thematik in dem vorliegenden Dokument habe ich versucht, von verschiedenen Blickrichtungen so zu bearbeiten, sodass sich ein geschlossenes Gesamtbild ergibt, welches auch für Unbeteiligte interessant und gut verständlich ist. Wenn das vorliegende Dokument in Zukunft auch für andere Personen und Institutionen von Nutzen ist und zu neuen Erkenntnissen beiträgt, würde ich mich freuen.

André Langer im November 2007

Inhaltsverzeichnis

ZUSAMMENFASSUNG ..... **V**

ABSTRACT ..... **VII**

DANKSAGUNG ..... **IX**

INHALTSVERZEICHNIS ..... **XI**

ABBILDUNGSVERZEICHNIS ..... **XV**

TABELLENVERZEICHNIS ..... **XVII**

ABKÜRZUNGSVERZEICHNIS ..... **XIX**

1. DIE VISION DES SEMANTIC WEB **21**

1.1. MOTIVATION 21
1.2. BESCHRÄNKUNGEN DES HEUTIGEN INTERNETS 23
1.3. DAS SEMANTIC WEB 27
1.4. ZIELSETZUNG DER ARBEIT 30
1.5. AKTUELLER STAND 32

2. GRUNDLEGENDE BETRACHTUNGEN **33**

2.1. WOZU PROJEKTMANAGEMENT? 33
2.2. BEGRIFFSDEFINITIONEN 35
2.2.1. Projekt *35*
2.2.2. Phase *36*
2.2.3. Prozess *37*
2.2.4. Workflow *38*
2.2.5. Aktivität *40*
2.2.6. Projektmanagement *40*
2.2.7. Prozessmanagement *41*
2.2.8. Workflowmanagement *42*
2.2.9. Groupware *42*
2.2.10. Knowledgemanagement *43*
2.3. MODELLIERUNGSANSÄTZE 43
2.3.1. Ereignisgesteuerte Prozessketten *45*
2.3.2. Flussdiagramme *45*
2.3.3. Petri Netze *46*
2.3.4. Unified Modeling Language *47*
2.3.5. Business Process Modeling Notation *48*
2.3.6. Zusammenfassung *48*
2.4. ANWENDUNGSSZENARIEN 49

2.4.1. Motivation *49*
2.4.2. Spaghetti kochen *49*
2.4.3. Abendessen mit Freunden *50*
2.4.4. Diplomarbeit schreiben *50*
2.4.5. Entwicklung einer neuen Webseite *51*
2.4.6. Workshop organisieren *52*
2.5. EVALUIERUNG GÄNGIGER SYSTEME 53
2.5.1. Herangehensweise *53*
2.5.2. Projektmanagementsysteme *55*
2.5.3. Workflowmanagementsysteme *62*
2.6. VERGLEICH VON WORKFLOW- UND PROJEKTMANAGEMENTSYSTEMEN 69

3. KONZEPTION EINES SEMANTIKBASIERTEN PROJEKTMANAGEMENTSYSTEMS 73

3.1. WAS BEDEUTET SEMANTIK? 73
3.2. WISSENSBESCHREIBUNG 76
3.3. ÜBERBLICK ÜBER TECHNOLOGIEN ZUR REPRÄSENTATION VON SEMANTIK 79
3.3.1. Motivation *79*
3.3.2. RDF *80*
3.3.3. N-Triples *81*
3.3.4. RDFS *82*
3.3.5. DAML+OIL *83*
3.3.6. OWL *83*
3.3.7. XML Topic Maps (XTM) *84*
3.3.8. XOL *85*
3.3.9. Layer-Architektur *85*
3.4. EINBETTUNG VON SEMANTIK IN (X)HTML-DOKUMENTE 86
3.4.1. Motivation *86*
3.4.2. SHOE *86*
3.4.3. Meta-Angaben *87*
3.4.4. Verlinkung auf externe RDF-Beschreibung *87*
3.4.5. Einbettung in Kommentarbereiche *88*
3.4.6. eRDF *88*
3.4.7. RDFa *88*
3.4.8. Microformats *89*
3.4.9. Vergleich von Microformats und RDFa *90*
3.5. NUTZBARE ONTOLOGIEN 92
3.6. ENTWURF EINER GEEIGNETEN ONTOLOGIE 95

4. IMPLEMENTIERUNG EINES PROTOTYPEN 101

4.1. FESTLEGUNG DER SYSTEMUMGEBUNG 101
4.2. SEMANTISCHE FRAMEWORKS 102

4.3. Wünschenswerte Funktionalitäten 104
4.4. Systemarchitektur 105
4.5. Lizenzmodell 110

5. PRAXISTEST 111

5.1. Installation 111
5.2. Performance 111
5.3. Bedienung 112
5.4. Testfälle 116

6. DISKUSSION 121

6.1. Beurteilung des Systemsentwurfs 121
6.2. Vergleich zu bisherigen Managementsystemen 125
6.3. Weiterentwicklungsansätze 128
6.4. Zukunft des Semantic Web 130

LITERATURVERZEICHNIS 133

INDEX 137

A. ANHANG 3

A.1. Getestete Projektmanagementsysteme 3
A.2. Getestete Workflowmanagementsysteme 15
A.3. Verwendete RDF Schemata zur Abbildung der Projekt- und Workflowmanagementdomain 23
A.4. Analyse und Spezifikation der zu entwickelnden Anwendung 33

Abbildungsverzeichnis

ABBILDUNG 2: GARTNERS HYPE CYCLE FÜR AUFSTREBENDE TECHNOLOGIEN 2006 30
ABBILDUNG 3: METHODEN DER PROZESSMODELLIERUNG NACH GADATSCH 44
ABBILDUNG 4: VERGLEICH GRUNDLEGENDER ELEMENTE VERSCHIEDENER MODELLIERUNGSANSÄTZE 47
ABBILDUNG 5: ENTWICKLUNG VON WORKFLOWBESCHREIBUNGSFORMATEN 49
ABBILDUNG 6: SCREENSHOT PHPROJECT 56
ABBILDUNG 7: SCREENSHOT DOUBLE CHOCO LATTE 57
ABBILDUNG 8: SCREENSHOT DOTPROJECT 58
ABBILDUNG 9: SCREENSHOT WEBCOLLAB 59
ABBILDUNG 10: SCREENSHOT PHPCOLLAB 60
ABBILDUNG 11: SCREENSHOT BASECAMP 62
ABBILDUNG 12: SCREENSHOT JAWE JAVA XPDL EDITOR 63
ABBILDUNG 13: SCREENSHOT BONITA 64
ABBILDUNG 14: SCREENSHOT IMIXS 65
ABBILDUNG 15: OSWORKFLOW 66
ABBILDUNG 16: SCREENSHOT RUNAWFE 67
ABBILDUNG 17: SCREENSHOT ORYX 68
ABBILDUNG 18: WISSENSPYRAMIDE NACH BODENDORF 75
ABBILDUNG 19: REUSABILITY-USABILITY TRADE-OFF PROBLEM 79
ABBILDUNG 20: SEMANTISCHES ARCHITEKTURMODELL 85
ABBILDUNG 21: NAIVES MODELL EINER PROJEKT- UND WORKFLOWMANAGEMENTONTOLOGIE 97
ABBILDUNG 22: VERZEICHNISHIERARCHIE DES PROJEKTMANAGEMENTSYSTEMS 106
ABBILDUNG 23: AUSSCHNITT AUS HTML-DATEI UND DARAUS ABGELEITETEN RDF STATEMENTS 107
ABBILDUNG 24: STARTSEITE VON SEMPROJ 113
ABBILDUNG 25: KONTAKTSEITE EINES NUTZERS MIT EXPORTMÖGLICHKEIT 113
ABBILDUNG 26: ZENTRALE ÜBERSICHTSSEITE 114
ABBILDUNG 27: PROJEKTEDITOR MIT GEÖFFNETEM EIGENSCHAFTENFENSTER 115
ABBILDUNG 28: TRANSFORMATION SEMANTISCHER METAINFORMATIONEN NACH RDF 124
ABBILDUNG 29: ATTRIBUTE EINER XPDL-BESCHREIBUNG 126
ABBILDUNG 30: "SPAGHETTI KOCHEN" ALS XPDL-BESCHREIBUNG 127
ABBILDUNG 31: EVOLUTION OF WEB TECHNOLOGIES 132

Tabellenverzeichnis

TABELLE 1: KATEGORISIERUNG VON WORKFLOWS NACH DER STRUKTUR 39
TABELLE 2: KRITERIENKATALOG FÜR DIE EVALUIERUNG VON WORKFLOW- UND PROJEKTMANAGEMENTSYSTEMEN 55
TABELLE 3: TYPISCHE OBJEKTEIGENSCHAFTEN VON PROJEKTEN 70
TABELLE 4: TYPISCHE OBJEKTEIGENSCHAFTEN VON AUFGABEN 71
TABELLE 5: VERGLEICH VON MICROFORMATS UND RDFA 91

Abkürzungsverzeichnis

AJAX	Asynchronous Javascript and XML
API	Application Programming Interface
BPMN	Business Process Modeling Notation
BSD	Berkeley Software Distribution
DAML	DARPA Agent Markup Language
DC	Dublin Core
DTD	Document Type Definition
EPK	Ereignisgesteuerte Prozesskette
FOAF	Friend of a Friend
GPL	General Public License
GRDDL	Gleaning Resource Descriptions from Dialects of Languages
HTML	Hypertext Markup Language
NS	Namespace
OIL	Ontology Inference Layer
OWL	Web Ontology Language
RDF	Ressource Description Framework
RDFS	RDF Schema
SGML	American Standard Code for Information Interchange
SHOE	Simple HTML Ontology Extensions
UML	Unified Modeling Language
URI	Uniform Ressource Identifier
URL	Uniform Ressource Locator
W3C	World Wide Web Consortium
WfMC	Workflow Management Coalition
WFMS	Workflow-Management-System
XHTML	eXtensible Hypertext Markup Language
XML	eXtensible Markup Language
XPDL	XML Process Definition Language
XSD	XML Schema Definition
XSL	eXtensible Stylesheet Language
XSLT	eXtensible Stylesheet Language Transformations

1. Die Vision des Semantic Web

"Things are only impossible until they're not." ~ Star Trek: The Next Generation - When The Bough Breaks, *Jean-Luc Picard, Season 1, 1988*

1.1. Motivation

Ein junger Unternehmensberater sitze an einem Dienstagmorgen in der Cafeteria eines namhaften Hotels irgendwo im Herzen von Deutschland. Während er mit einem Löffel in der linken Hand die Tasse Kaffee vor sich auf dem Tisch umrührt, hält er in der rechten Hand bereits seinen PDA, um sich einen Überblick über die Termine des kommenden Tages zu verschaffen. Nach einem Klick auf „*heutige Termine anzeigen*" liefert das System die Beschreibung: „*Sie haben heute um 10.00 Uhr ein Treffen mit Dr. Müller in der Ulmenstraße 5. Um 15.30 Uhr ist eine Telefonkonferenz mit der Arbeitsgruppe Asterisk angesetzt. 17.00 Uhr haben Sie einen Termin mit Herrn Holger Schmidt vorgemerkt. Um 18.20 Uhr schließlich startet ihr Flug nach Grenoble.*" Der Berater überlegt, dass es besser wäre, den Termin mit *Herrn Schmidt* zu verlegen, um rechtzeitig auf dem Flughafen zu sein. Er verschiebt den Termin um zwei Tage, was durch das System zunächst abgelehnt wird mit der Begründung, dass sich *Herr Schmidt* zu dieser Zeit auf einer Dienstreise befindet. Nach einer Bestätigung legt das System das Treffen auf den nächstmöglichen Termin und versendet an *Herrn Schmidts* Emailadresse eine entsprechende Nachricht. Die nächste Anfrage, worum es bei der Telefonkonferenz geht, beantwortet das System mit einer Meldung „*Die Firma Meier möchte diskutieren, was ein Umstieg auf SCCP für Vorteile bringt.*" *SCCP* ist dem Berater unbekannt, worauf ihm das System den Begriff definiert[1] und weitere Informationen liefert. Nach dem Frühstück macht sich der Berater auf zu seinem ersten Termin, lässt von dem System automatisch ein *Taxi* rufen und fragt auf dem Weg nach draußen noch schnell ab, wann und wo er *Dr. Müller* letztmalig getroffen und wo sein Geschäftspartner den letzten Urlaub verbracht hat.

Das skizzierte Szenario scheint auf dem ersten Blick in ferner Zukunft zu liegen. Derartige Interaktionen zwischen Mensch und Maschine werden häufig im Bereich der Science-Fiction angesiedelt. Zu allgegenwärtig scheinen eigene Erfahrungen, wie aufwändig es sein kann, eine Suche nach spezifischen Daten im Internet mit Computerunterstützung durchzuführen. Jegliche Informationen, welche nicht explizit in einer Organizer-Software gespeichert sind und entsprechend angezeigt werden können, scheinen für einen Rechner nicht auswertbar. Zu groß wäre der Suchaufwand in der Informationsfülle des heutigen Internets – ganz abgesehen von der Fragestellung, wie ein Rechner Beziehungen zwischen Informationen herstellen solle, Mehrdeutigkeiten auflösen und eine auf die Fragestellung zugeschnittene, für den Menschen verständliche und vereinfachte Ausgabe zurückliefern kann.

[1] Skinny Client Control Protocol, ein von Cisco Systems Inc. entwickeltes, proprietäres Protokoll zur Abhaltung von Telefonkonferenzen über VoIP in Echtzeit

Dennoch ist eine derartige Anwendung heutzutage vorstellbar und nicht länger Fiktion. Der Traum von intelligenten Maschinen besteht schon seit langer Zeit. Beschränkt man sich nur auf Entwicklungen mit Bezug zur modernen Rechentechnik, so prägte Alan Turing 1950 erstmals die Vorstellung von intelligenten Maschinen [Tur50]. Der sich daran anschließende Optimismus unter KI-Forschung bis Ende der 60er Jahre wurde durch eine Phase der Ernüchterung beendet, als man zunehmend die Beschränkungen grundlegender Konzepte und der darauf basierenden Algorithmen erkannte. Während in den Siebziger und Achtziger Jahren neue Erfolge bei der Entwicklung kommerzieller Expertensysteme erzielt worden, lag der Fokus bei der Verbreitung des World Wide Webs nach Einführung des Hypertext-Konzepts durch Sir Tim Berners-Lee im Jahr 1989 zunächst auf der öffentlichen Bereitstellung von Informationen in Dokumenten, welche auch für Endanwender einfach zu realisieren sein sollte. Die Hypertext Markup Language (HTML) als Anwendung der Standard Generalized Markup Language (SGML), mit der Dokumentstrukturen syntaktisch beschrieben werden konnten, schuf dabei die Grundlage für einen Dokumenttyp, über den mithilfe einiger zusätzlicher Auszeichnungen zur Angabe von Textformatierungen alle Dokumentinformationen in einem unitären Dokumentformat in einer einheitlichen Struktur gespeichert, verbreitet und zur Darstellung bereitgestellt werden konnten. Nutzer wurden dadurch in die Lage versetzt, Textinhalte einfach formatieren und mit anderen Dokumenten und Multimediadateien verknüpfen zu können. Verbunden mit der einfachen Erlernbarkeit und Einsetzbarkeit von HTML durch Privatanwender war jedoch die Einschränkung, einen festen Satz an vordefinierten Auszeichnungselementen („Tags“) vorzugeben, welchen eine explizite Bedeutung zugeordnet wurde. Im Bezug auf die Rücktransformation von Dokumentinhalten in Informationsstrukturen stellte dies ein Problem dar, da Informationen über die Natur der dargestellten Inhalte nicht gespeichert wurden (es konnten bzw. sollten keine eigenen Auszeichnungselemente hinzugefügt werden), sondern HTML in erster Linie die Dokumentdarstellung unterstrich. Durch den einfachen Syntax und die schnelle Bereitstellung von Entwicklungswerkzeugen (WYSIWYG-Editoren) verbreitete sich das Hypertext-Konzept binnen kurzer Zeit und fand breite Akzeptanz[2], dennoch wurden besonders im Bereich der rechnergestützten Dokumentverarbeitung Anforderungen immer wichtiger, nicht nur Darstellungsinformationen sondern auch syntaktische Dokumentstrukturinformationen in einem universellen, einheitlichen Format speichern und übertragen zu können. Als Folge davon wurde die eXtensible Markup Language (XML) entwickelt, welche sich bis heute als Austauschformat in unterschiedlichsten Anwendungsbereichen etabliert hat.

[2] An manchen Stellen wird in HTML dadurch sogar der wohl bedeutendste Schritt gesehen, der erst das Internet in der heutigen Realisation ermöglichte, da die Auszeichnungssprache weltweit akzeptiert wurde [Lac05, p. 1]. Vielfach wurde versucht, weitere Standardsprachen („lingua franca des Internets“) für andere Anwendungsbereiche einzuführen, die jedoch alle mehr oder weniger an unterschiedlichen Auffassungen verschiedener Organisationen und Länder gescheitert sind oder modifiziert werden mussten.

HTML und XML (und weitere Konzepte wie CSS) widersprechen dabei nicht einander, sondern können sich gegenseitig ergänzen und ineinander überführt werden (XSLT) , da beide der gleichen Sprachfamilie entstammen und mit der Spezifikation von XHTML1.0 dies sogar eine Untermenge von XML mit einem dedizierten Anwendungsfeld darstellt. Je nach Anwendungsfall mit Schwerpunkt auf Layout, Struktur oder Inhalt wird man sich für eine konkrete Umsetzungsmöglichkeit entscheiden, wodurch eine gewisse Trennung in einen layoutorientierten oder strukturorientierten Ansatz vorhanden ist. Trotz dass XML im letzteren Fall schon den Anschein erweckt, Informationen über Dokumentinhalte perfekt abbilden zu können, reichte aber auch dies nicht aus, um Inhalte für einen Computer verständlich werden zu lassen, da damit zwar eine syntaktische Korrektheit sichergestellt werden konnte, nicht jedoch eine semantische Korrektheit.[3]

1.2. Beschränkungen des heutigen Internets

Das World Wide Web ist im Wesentlichen ein Medium zur Veröffentlichung und Distribution von Informationen. Ende Dezember 2006 waren weltweit 120 Millionen Domains registriert. Pro Monat kommen statistisch 3,2 Millionen neue Domains hinzu. [Ver07]. Die Anzahl der tatsächlich im Internet vorhandenen Dokumente kann nur geschätzt werden, wobei Millionen privater Homepages mit wenigen Unterseiten und Webseiten großer Unternehmen mit womöglich mehr als 100.000 Dokumenten gleichermaßen ins Gewicht fallen. Eine Hochrechnung basierend auf Daten des Archivierungsdienstes www.archive.org im Frühjahr 2006 geht von einer kumulativen Summe von insgesamt 55 Milliarden (1 Petabyte) Dokumenten seit 1996 aus mit einem Zuwachs von 20 Terabyte an Daten pro Monat. Andere Statistiken sehen diese Schätzungen als zu pessimistisch an und sprechen von einer zwei Drittel höheren Dokumentenanzahl. [Wen06] Diese Zahl von über 160 Milliarden Dokumenten im World Wide Web basiert auf der Annahme, dass aktuelle Suchmaschinen real nur ca. 35 % aller tatsächlich vorhandenen Dokumente indiziert haben (Abbildung 1). Im Umkehrschluss bedeutet dies, dass ein Großteil potentiell relevanter Informationen entweder gar nicht oder nur indirekt gefunden werden kann. Selbst die Informationen, auf die durch Suchmaschinen öffentlich zugegriffen werden kann, existieren unabhängig voneinander. Abgesehen von Techniken wie Web Scraping oder explizit angebotenen Web Services ist der Datenbestand einer Webapplikation auf die eigene Anwendung bzw. Domain begrenzt, über die der zuständige Entwickler zusätzliches Hintergrundwissen besitzt. Könnte man den Inhalt der Milliarden Dokumente miteinander verknüpfen, daraus Informationen abrufen, den Inhalt und Wahrheitswert überprüfen und mitunter sogar neue Informationen ableiten, wären ein Informationsgewinn und Anwendungen denkbar, die derzeit noch nicht vorstellbar sind, da das spezifische Suchen, Vergleichen und Ableiten von Informationen bisher menschliche Interaktion erfordert.

[3] Breitman, Casanova und Truszkowski bezeichnen in diesem Zusammenhang das heutige Internet als „Syntactic Web“ im Gegensatz zum angestrebten „Semantic Web“ [Bre07, p.5]

Abbildung 1: Die Sicht einer heutigen Suchmaschine auf das WWW, Quelle: http://www.foi.se/FOI/templates/Page____4070.aspx

Ein weiteres Problem stellt darauf aufbauend die Art und Weise der Suche nach Informationen dar. Moderne Algorithmen, die in heutigen Suchmaschinen zur Anwendung kommen, sind zwar hochkomplex und haben sich in den vergangenen zwei Jahrzehnten qualitativ kontinuierlich weiterentwickelt, doch ist das Grundkonzept dahingehend gleich geblieben, dass alle Suchanfragen in der Regel auf einzelnen Schlüsselwörtern basieren und deren Vorkommen im (Volltext-)Index des Suchmaschinendatenbestandes überprüft wird. Die Konsequenz daraus ist, dass zwar syntaktische Analysen durchgeführt können und nach ähnlichen Wörtern gesucht werden kann, in aller Regel auch die Anzahl des gemeinsamen Vorkommens der Schlüsselwörter im Zieldokument sowie die Popularität der Seite eine Rolle spielt, doch bleibt die Bedeutung der Anfrage in aller Regel verborgen. So ist aus einem einzelnen Schlüsselwort wie etwa *Flügel* nicht ableitbar, ob der Nutzer *eine Webseite sucht, die Musikinstrumente anbietet, eine Anleitung für eine mechanische Konstruktion sucht,* oder *in ein zoologisches Lexikon schauen* möchte. Dementsprechend bieten heutige Suchmaschinen in einer Übersicht häufig die „geeignetsten“ Treffer an, worunter derartige Begriffsdeutungen zumeist miteinander vermischt sind. In manchen Situationen hilft es, mehrere Schlüsselwörter in die Suchanfrage aufzunehmen (*+wie +spielt +man +auf +einem +Flügel*), wodurch Dokumente gefunden werden könnten, die die Begriffe *spielen* und *Flügel* enthalten, doch wäre es intuitiver, die Frage in dieser Form direkt an einen Suchdienst stellen zu können, der diese Frage als Ganzes versteht und nur passende Resultate zurückliefert (auch solche, in denen das Wort *Flügel* nie erwähnt wird, dafür aber vielleicht von einem *Klavier* gesprochen wird).

Dass dies technisch nicht trivial umzusetzen ist, liegt wie in Abschnitt 1.1. bereits kurz beschrieben darin begründet, dass Dokumentinhalte im World Wide Web größtenteils im (X)HTML – Format vorliegen. Der Fokus auf einer einfachen Layoutbeschreibung brachte jedoch den Nachteil mit sich, dass die Beschreibung des Inhalts von HTML-Dokumenten nur zweitranging betont wurde (Meta-Tagging). Die Folge davon ist, dass die entstandenen HTML-Dokumente zwar in ihrer Struktur durch einen Computer gut verarbeitbar sind, der Inhalt computergeschützt aber nicht trivial Konzepten der realen Welt zuzuordnen ist (die Tags zur Auszeichnung können entfernt werden, der resultierende Text an sich ist strukturlos und kann nur mithilfe von String-Operationen verarbeitet werden), was wiederum zu dem angesprochenen Problem führt, bestimmte Sachverhalte bei einem Indexierungsvorgang semantisch voneinander unterscheiden zu können.

Einige Suchmaschinen verwenden dafür zwar Algorithmen, welche den Inhalt bestimmter Textmarken wie von (*h1*-) Überschriften stärker wichten und als Beschreibung des weiteren Dokumentinhalts ansehen, doch entspricht jeder weitere Schritt im Grunde genommen einer syntaktischen Volltextsuche.

Um eine einfach zu benutzende, stärker inhaltszentrierte Verarbeitung zu ermöglichen, wurde die eXtensible Markup Language entwickelt. Durch XML wird es möglich, Daten im Text logisch voneinander zu trennen und separat beliebig auszuzeichnen (zu benennen). Ebenso können syntaktische Regeln für die ausgezeichneten Daten definiert werden (via DTD/XSD). Die rechnergestützte Verarbeitung der Daten wird damit vereinfacht, da auf einzelne Dokumentinhalte nach einem standardisierten Schema gezielt zugegriffen werden kann. Nichtsdestotrotz blieben einige Probleme bestehen, die auch mithilfe von XML nicht gelöst werden können, um Dokumentinhalte durch ein Computersystem anhand der Bedeutung der Inhalte verarbeiten zu lassen. Diese seien nachfolgend genannt:

- Die enthaltenen Informationen in einem XML-Dokument sind für einen Rechner weiterhin Zeichenketten ohne explizite Bedeutung. Dokumentinhalte sind zwar mithilfe von XML syntaktisch ausgezeichnet und orientieren sich mehr am Textinhalt als an der Darstellung des Inhalts, doch ist die konkrete Bezeichnung der ausgezeichneten Bereiche für den Computer an sich beliebig wählbar und nur durch den Menschen deutbar.
- Mithilfe definierter Datentypen in XML Schema ist es zwar möglich, Abhängigkeiten zwischen einzelnen Elementen zu definieren (*eine Adresse besteht aus Name, Straße, Postleitzahl, Ort*), diese sind jedoch ebenfalls nur syntaktischer Natur
- XML-Dokumente sind zwar strukturell alle auf gleiche Art und Weise verarbeitbar, doch bedeutet dies nicht, dass eine Aussage über die reale Welt in einem XML-Dokument eindeutig auf eine bestimmte Art und Weise beschrieben werden kann. Vielfach kann die gleiche Information mithilfe unterschiedlicher Attribut- und Knotenschachtelungen in einem Dokument abgebildet werden. Für eine rechnergestützte Verarbeitung von Informationen sind jedoch eindeutige Repräsentationen für Aussagen mit gleichem Inhalt zwingend erforderlich.
- Syntaktische Strukturinformationen (DTD/XSD) sind lokal auf ein Dokument beschränkt. Weitere XML-Dokumente können zwar eingebunden werden (XInclude), welche dann jedoch zu der verwendeten Schemadefinition kompatibel sein müssen. Weitere Anforderungen an Beziehungen zwischen verschiedenen Strukturen (beispielsweise eine *ist ein* – Relation) können nur unbefriedigend abgebildet werden.

- XML ist ein in der Praxis gut einzusetzendes Austauschformat, welches eine Kompatibilität zwischen mehreren Institutionen schafft, doch fehlen für eine domainübergreifende inhaltliche Auswertung einige Möglichkeiten. Eine Suchmaschine kann in einem Dokument nach einem gängigen Tag *<title></title>* suchen, wobei bei einem positiven Suchergebnis nicht feststeht, ob der Elementinhalt wirklich den Titel der Webseite angibt, oder vielleicht den Titel eines darin behandelten Buches oder vielleicht etwas ganz anderes. Ebenso ist bei Nichtauffinden des Elements nicht gesagt, dass nicht die Titelinformation unter einer anderen Bezeichnung in dem Dokument existiert.
- Die Extraktion von Informationen aus einem XML-Dokument geschieht in XML-Anwendungssoftware nach einem festen Schema. (*Um den Titel einer Webseite zu ermitteln, suche nach einem Element title und nutze die darin enthaltenen Daten*) Änderungen an den XML-Markupdaten ziehen direkte Änderungen im Programmcode der Anwendungssoftware nach sich.
- Zuletzt sollte das Aufwand-Nutzen-Verhätlnis für die Bereitstellung von Informationen in einem XML-Dokument in die Überlegungen einbezogen werden. Der einzelne Endanwender wird nur in seltenen Fällen private Informationen als XML-Datei öffentlich bereitstellen, da dafür in vielen Fällen das Wissen fehlt, oder es doppelten Aufwand erfordert, sowohl die Informationen in einer HTML-Datei mit Darstellungsangaben zu versehen und diese zusätzlich in einer XML-Datei für eine maschinelle Verarbeitung zu hinterlegen. In einigen Fällen macht dies Sinn (bspw. FOAF, vCards), stellt in jedem Fall jedoch einen Zusatzaufwand auf Nutzerseite dar. Techniken wie XSLT existieren, um Inhalte einer XML-Datei in einem einzelnen Prozess mit Styleangaben zu versehen und zur Darstellung zu bringen, doch können diese die klassische Bereitstellung von Informationen auf einer Webseite nicht ersetzen. Denn letztendlich ist der eigentliche Erfolg des Internets darauf zurückzuführen, dass Nutzer bereit sind, eigene Inhalte zu veröffentlichen, da sie wiederum Inhalte von anderen Nutzern nutzen können und sich die Zeit, die sie in das Veröffentlichen eigener Dokumente investieren, im Vergleich zu den Vorteilen und Möglichkeiten, die im Internet geboten werden, rentiert.

Während der vergangenen 18 Jahre hat sich das World Wide Web rasant weiterentwickelt. Einige Wissenschaftler sprechen bei der „*Evolution des Internets vom größten technologischen Fortschritt der Geschichte*“ [Lac05, p. 2]. Inzwischen existieren Applikationen, welche vor Jahren nicht denkbar waren – angefangen bei der Möglichkeit, Internetseiten dynamisch mithilfe von Programmiersprachen wie PHP, Java oder .NET erstellen zu können bis hin zu Web-Services und Technologien wie AJAX, welche im Rahmen der Web 2.0 – Entwicklung an Bedeutung gewonnen haben.

Was die gerade geschilderten Schwächen des heutigen Internets angeht, so scheint es möglich, dass auch dafür neue Lösungen gefunden werden können, mit deren Hilfe der Informationszugriff durch zukünftige Anwendungen in einer noch nicht da gewesenen Art und Weise ablaufen könnte. Dazu ist im Grunde genommen kein vollkommen „neues" Internet mit anderen Basistechnologien nötig, da die eXtensible Markup Language an sich dafür flexibel genug ist und dafür in den vergangenen Jahren eine Vielzahl an ausgereiften Werkzeugen entwickelt wurden, die auch in Zukunft nutzbar sein sollten. Dass semantikbasierte Anwendungen bisher noch nicht verbreitet sind, ist primär auf eine unbefriedigende Informationsrepräsentationen zurückzuführen, die Inhalte in einer allgemeingültigen, maschinenverständlichen Form auszeichnen können. Der leitende Forscher zur Entwicklung der DARPA Agent Markup language (DAML) schilderte dies wiefolgt: „*The current web is a victim of its own success. […] The web must be terraformed to allow automated software agents to explore its expanses*" [Lac05, p.6].

1.3. Das Semantic Web

Erste Bestrebungen, Inhalte besser beschreiben und über Anwendungsgrenzen hinaus für automatische Verarbeitungen bereitstellen zu können, gab es bereits 1995, wo mithilfe des XML-basierten Meta Content Frameworks (MCF) versucht wurde, Informationsstrukturen einheitlich zu beschreiben. Eine drei Jahre später daraus entwickelte, allgemeinere Methode zur Beschreibung von Informationen – das Ressource Description Framework (RDF) – ermöglichte es „*to say anything about anything*", indem mithilfe von Metadata Aussagen über Objekte getroffen werden konnten.

Der Begriff eines Semantik-basierten Internets schließlich wurde maßgeblich durch die Ideen von Sir Tim Berners-Lee geprägt. So schilderte Berners-Lee in *Weaving the Web* [Ber99] : „*I have a dream for the Web ... and it has two parts. In the first part, the Web becomes a much more powerful means for collaboration between people. […] In the second part of the dream, collaborations extend to computers. Machines become capable of analyzing all the data on the Web - the content, links, and transactions between people and computers. A 'Semantic Web', which should make this possible, has yet to emerge, but when it does, the day-to-day mechanisms of trade, bureaucracy, and our daily lives will be handled by machines talking to machines, leaving humans to provide the inspiration and intuition.*"

Seitdem wird weltweit geforscht, um diese Vision Realität werden zu lassen. Sowohl in den USA, als auch in Europa wurden erste Ansätze entwickelt, um Inhalte so zu beschreiben, dass sie sowohl von Menschen als auch Computern verstanden werden können. Das United States Department of Defence entwickelte dazu im Jahr 2000 die DARPA Agent Markup Language, während in Europa parallel dazu an einer expliziten Repräsentation von Semantik durch die Ontology Inference Language (OIL) geforscht wurde. Beide Konzepte lieferten wichtige Erkenntnisse für weitere Forschungsarbeiten und wurden schließlich zu DAML+OIL vereint.

Die Web Ontology Group des World Wide Web Consortiums (W3C) schließlich versuchte im Februar 2004 unter der Leitung durch Sir Tim Berners-Lee eine semantische Auszeichnungssprache für zukünftige Anwendungen zu standardisieren. Aus der Kombination verschiedener Vorschläge ging schließlich die Web Ontology Language (OWL) hervor, die inzwischen eine breite Unterstützung gefunden hat. Dennoch existieren weitere semantische Sprachen und Dialekte und es scheint bisher noch unklar, welches Konzept sich auf Dauer durchsetzen wird, auch wenn die Konzeption hinter OWL schlüssig scheint und zunehmend anerkannt wird.

Insgesamt gesehen ist die Forschung an einem Semantic Web noch recht neu. Grundlagenforschung wurde zwar bereits vor Jahrzehnten im Bereich der Künstlichen Intelligenz betrieben, doch ist die Übertragung der Konzepte auf das Internet nicht einfach, weswegen zunächst viel über theoretische Grundlagen, Begrifflichkeiten, Modelle und Zielsetzungen des Semantic Web diskutiert wurde. Der Begriff des „Semantic Web“ ist dahingehend zu einer Art buzzword geworden, von dem jeder spricht und eine eigene Vorstellung dazu hat, was es umfasst. Den Begriff des Semantic Web eindeutig zu definieren ist daher schwierig und bisher hat sich auch keine eindeutige Definition durchgesetzt. Mögliche Ansätze sind:

- *“The Semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation”* [Ber01]
- “[The Semantic Web is an] *International effort to make the vast data resources of the World Wide Web available in a format amenable to automated processing by intelligent agents and other computer programs.* […] *However, most Semantic Web technology will be transparent to users”* [Pas04]
- *“The Semantic Web will give you software agents that will tackle your daily needs quietly, effectively, even with common sense, finding the information they need and negotiating with other agents on your behalf. Or, the Semantic Web will be a little more of what we have now – a little faster, a little smarter, with more complex software. Or, the Semantic Web is all hype”* [Pas04]
- *“The Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries. It is a collaborative effort led by W3C with participation from a large number of researchers and industrial partners. It is based on the Resource Description Framework”* (W3C, http://www.w3.org/2001/sw/)
- “*The Semantic Web* [is] *a vast object-oriented distributed database with machine-understandable schemas. Information representations will be distributed on geographically-distributed servers controlled by diverse authors of content. Just as its value grew as more sites were linked, the Semantic Web will become more valuable as information representations are connected”* [Pas04]

- *"The aim of the Semantic Web is to turn the web from a large hyperlinked book into a large interlinked database"* (Semantic Web Advanced Development for Europe, http://www.w3.org/2001/sw/Europe/)
- "*The Semantic Web concept is to do for data what HTML did for textual information systems: to provide sufficient flexibility to be able to represent all databases and logic rules to link them together to great added value*" (proposal for a Semantic Web Logic Language, W3C 2000, http://www.w3.org/2000/01/sw/DevelopmentProposal)
- "*The vision of Semantic Web is to let computer software relieve us of much the burden of locating resources on the Web that are relevant to our needs and extracting, integrating, and indexing the information contained within*" [Cra01]

Die vorgestellten Definitionsansätze lassen sich prinzipiell in zwei Kategorien einteilen. Entweder wird versucht, das Konzept des Semantic Web anhand des Verhaltens darin vorstellbarer Applikationen zu beschreiben („*data linked together to provide new services*"), oder es wird näher auf die technische Realisierungsbasis eingegangen, inwieweit sich das Semantic Web vom heutigen Internet unterscheiden soll („*requires joining together data that is located in independent application domains*"). Insgesamt wird ersichtlich, dass der Begriff des Semantic Webs nicht einfach abzugrenzen ist und dessen inhaltliche Deutung sich auch in der kommenden Zeit noch ändern kann.

Im Folgenden wird unter dem Begriff des Semantic Webs ein zukunftsträchtiger Ansatz gesehen, der sich an folgender Definition orientiert:

Definition 1: Semantic Web
Das Semantic Web stellt ein Konzept dar, bestehende Internetinhalte um Informationen zu ergänzen, mit welchen autonome Softwaresysteme in die Lage versetzt werden, den Sinngehalt von Daten eigenständig extrahieren, aufzubereiten und mit anderen Informationsquellen domainübergreifend kombinieren zu können mit dem Ziel, aus verteilten Wissensbasen qualitativ hochwertige Ergebnisse zu extrahieren, wozu bisher die Interaktion mit einem menschlichen Nutzer nötig war.

1.4. Zielsetzung der Arbeit

Das Ziel der vorliegenden Diplomarbeit ist es, die in den vergangenen Jahren entwickelten Technologien und Konzepte des Semantic Web auf Ihre praktische Anwendbarkeit zu testen. Die praktische Umsetzbarkeit wird an der Realisierung eines prototypischen Webapplikation getestet, welche grundlegende Projekt- und Workflowmanagementaktivitäten ermöglichen soll. Diese Anwendungsdomain ist momentan im Kontext des kollaborativen Arbeitens verbunden mit einer semantikbasierten Umsetzung hochaktuell, gleichzeitig aber auch so gut erforscht, dass auf eine Reihe fester Konzepte und Begrifflichkeiten zurückgegriffen werden kann, welche sich semantisch gut beschreiben lassen. Der Hype-Cycle in Abbildung 2 veranschaulicht die momentane Präsenz des Themas.

Alle zu verarbeitenden Informationen in dem zu entwickelnden System werden in einem Pool aus verteilten (Web-)Dokumenten bereitgestellt, wobei die darin enthaltenen Daten mit zusätzlichen Informationen zur Beschreibung der Semantik versehen sind. Das gesamte System wird darauf ausgelegt, den kollaborativen Gedanken des Semantic Web bestmöglich umzusetzen.

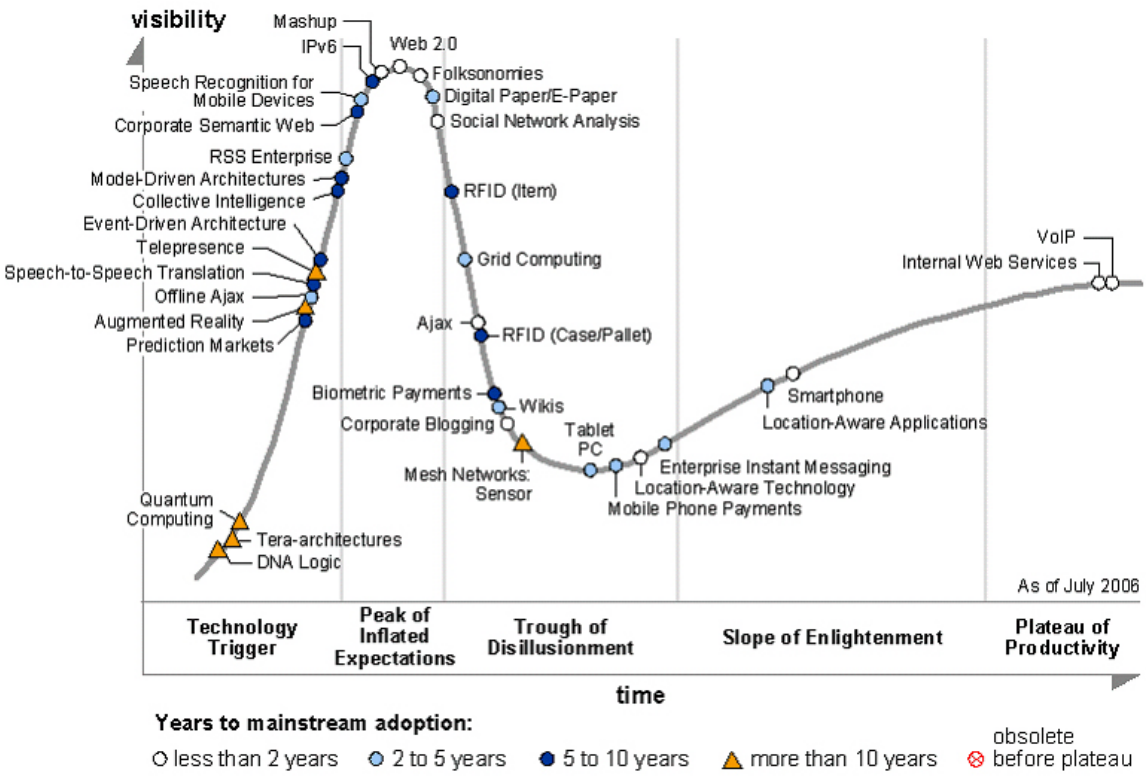


Abbildung 2: Gartners Hype Cycle für aufstrebende Technologien 2006, Quelle: http://www.geospatialsemanticweb.com/wp-content/uploads/2006/08/hypecycle2006.jpg

Es soll demonstriert werden, welche Möglichkeiten existieren, eine Problemstellung in einem lauffähigen System praktisch abzubilden, modellrelevante Informationen aus dazugehörigen Webdokumenten zu extrahieren und diese so aufzubereiten, dass sie einen Mehrwert für den Nutzer liefern können. Die Grundlagen von Projektmanagementsystemen sind in den letzten Jahrzehnten bereits gut erforscht worden, wodurch eine Vielzahl an Vergleichssystemen zur Verfügung steht, deren Stärken und Schwächen zunächst untersucht werden können und aus denen ein entsprechendes Grundvokabular abgeleitet werden kann. Das zu entwickelnde System hat schließlich der Anforderung zu genügen, durch eine einfache Bedienung den Menschen bei der täglichen Arbeit zu unterstützen, was im Kern das Grundanliegen der Informatik ist (vgl. [Wah06]).

Nachdem in diesem Einführungskapitel dazu zunächst ein Überblick über den aktuellen Forschungsstand gegeben wurde, sollen in Kapitel 2 zunächst grundlegende Begriffe zur Realisierung eines Projektmanagementsystems geklärt sowie ein Überblick über im Einsatz befindliche Managementsysteme gegeben werden. Besonderer Wichtigkeit kommt dabei der Definition eines Grundvokabulars zu, welches für alle Beteiligten wohl unterscheidbare Konzepte umfasst. In Kapitel 3 schließlich werden die Ideen des Semantic Webs aus Kapitel 1 mit den Grundlagen des Projektmanagements aus Kapitel 2 verknüpft und ein Konzept erarbeitet, wie ein Semantic Web – basiertes System zur Unterstützung von Workflow- und Projektmanagement aussehen könnte. Kapitel 4 beschreibt die konkrete Umsetzung des Konzeptes basierend auf einem ersten Applikations-Prototypen in C# / ASP.NET. Nachdem in Kapitel 5 die Funktionalität und Bedienung im praktischen Einsatz vorgestellt wird, werden in Kapitel 6 die gewonnenen Erkenntnisse diskutiert und Vergleiche zu herkömmlichen Projektmanagementsystemen angestellt. Abschließend wird ein Ausblick auf mögliche Weiterentwicklungsmöglichkeiten des Prototypen gegeben und eine Vermutung aufgestellt, welche Entwicklungen im Rahmen der Semantic Web – Bewegung in den kommenden Monaten zu erwarten sind und welche Probleme dabei mitunter noch gelöst werden müssen, welche in dieser Diplomarbeit aufgefallen sind. Zu betonen ist dabei, dass in dieser Arbeit die Entwicklung eines akademischen Prototyps im Mittelpunkt steht, der grundlegende Funktionen bereitstellt, der die Funktionalitäten von herkömmlichen, über mehrere Jahre professionell entwickelten Systemen jedoch nicht erreichen kann und möchte.

1.5. Aktueller Stand

Das Angebot an Literatur zu klassischem und modernen Projektmanagement ist nahezu unüberschaubar. Allein eine Suche bei Amazon.com nach dem Begriff „*Project Management*“ liefert knapp 32.000 Treffer zurück (durchgeführt am 27.07.2006). In der gleichen Größenordnung liegen Veröffentlichungen zu dem Thema Process- oder Workflow-Management sowie Knowledge Management. Im Gegensatz zu der interdisziplinären Ausrichtung dieser Diplomarbeit liegt der Schwerpunkt der Mehrheit aller erhältlichen Publikationen jedoch im wirtschaftswissenschaftlichen Bereich. Daraus entstandene Projekt- oder Workflowmanagementsysteme sind in großer Vielzahl sowohl in kommerziellen Versionen vorhanden als auch frei verfügbar. Eine entsprechende Übersicht bietet beispielsweise http://www.softguide.de/software/projektmanagement.htm. Die informationstechnische Umsetzung eines derartigen Systems wird unter anderem in [Kir94] und [Hab98] behandelt. Neuere Arbeiten aus dem Bereich der Informatik konzentrieren sich insbesondere auf die Modellierung von Prozessen mittels UML [Pet99] und anderen deskriptiven, XML-basierten Ansätzen [Hun06]. Möglichkeiten, Ontologie-basierte Systeme zum Workflow- und Projektmanagement einzusetzen, werden von vielen Autoren aufgezeigt, jedoch bisher selten vertieft [Pel06]. Besonderer Bedeutung kommt dabei der Wissensvernetzung in Organisationen [Sch06] und dem kollaborativen Wissensmanagement [Sur06] zu.

Nachdem vor allem im Bibliotheksbereich frühzeitig die Auszeichnung von Informationen erprobt wurde [Sch00], finden sich seit einiger Zeit eine Reihe weiterer Anwendungen im Internet, welche mithilfe eines semantischen Markups Informationen bereitstellen oder vorhandene Informationen aufbereiten und aggregieren (als Beispiel sei unter anderem der Suchdienst www.kartoo.com genannt). Im Bereich des Wissensmanagements wurde frühzeitig versucht, die Idee des Semantic Web auf Wiki-artige Systeme zu übertragen und daraus Semantische Wikis mit neuartigen Funktionen zu entwickeln (siehe http://wiki.ontoworld.org/wiki/Category:Semantic_wiki und http://platypuswiki.sourceforge.net/) . Die Weiterentwicklung zu einem vollwertigen System zur Unterstützung von Wissensmanagement wurde unter anderem in dem Ike Wiki – Projekt [Ike06] angestrebt, die aktuell verfügbare Testumgebung lässt jedoch keinen Schluss auf die Weiterentwicklung dieser Idee zu. Aufgrund des großen Interesses im Anwenderbereich an einem derartigen System zur Unterstützung von Wissens- und Projektmanagement ist die Forschung an diesem Thema hochrelevant. Neue Veröffentlichungen zeigen dabei unter anderem auf, wie eine Kooperation zwischen Projekt- und Workflowmanagementsystemen aussehen könnte [Bau04]. Darin wird vorgestellt, inwieweit Projektmanagement und Workflowmanagement sich von unterschiedlichen Blickpunkten um die Realisierung und Überwachung von Abläufen kümmern. Projektmanagement könnte dabei auf einer abstrakteren Ebene ablaufend angesehen werden als Workflowmanagement. Mit einem semantikbasierten Ansatz könnten diese beiden Systemwelten nun auf inhaltlicher Ebene vereint werden, was nachfolgend in dieser Arbeit demonstriert werden soll.

2. Grundlegende Betrachtungen

„Sag mir, wie ein Projekt beginnt und ich sage Dir, wie es endet." ~ Gero Lomnitz, Institut für praktische Psychologie und Organisationsberatung, Köln

2.1. Wozu Projektmanagement?

Sobald mehrere Menschen zusammenarbeiten, um ein gemeinsames Ziel zu erreichen, spricht man umgangssprachlich von einem Projekt. Dementsprechend sind Methoden zur Aufteilung von Aufgaben und zur Überwachung von Arbeitsabläufen schon seit langer Zeit bekannt. Bereits in der Antike sollen große Vorhaben wie der Bau der ägyptischen Pyramiden oder der Chinesischen Mauer im Kontext eines Projektes durchgeführt worden sein. [Möl00]

Die Vorstellungen des heutigen Projektmanagements gehen bis in das 18. Jahrhundert zurück und wurden maßgeblich durch den aus Schottland stammenden Ökonomen Adam Smith (1723 - 1790) geprägt. Besondere Bedeutung kommt dabei dem Buch „Der Wohlstand der Nationen" (1776) zu, in dem Smith Möglichkeiten aufzeigt, die Produktivität von Arbeitskräften um ein Vielfaches zu steigern und damit gleichzeitig Produktions- und Warenkosten maßgeblich zu senken. Im anbrechenden Zeitalter der Industriellen Revolution wurden darauf aufbauend verstärkt neue Firmenstrukturen und standardisierte Betriebsabläufe geschaffen. Ständig wiederkehrende Handlungsabläufe wurden im Zuge dieser Standardisierung identifiziert, in abgegrenzte Zuständigkeitsbereiche aufgeteilt und die Ergebnisse dokumentiert, sodass sie von hoch spezialisierten Arbeitskräften mit klar formulierten Aufgabenbereichen ausgeführt werden konnten und etwaige Abweichungen oder Probleme sofort erkannt wurden. Der gesamte Arbeitsablauf wurde durch diese Optimierungen effizienter, zuverlässiger, sicherer und vor allem vorhersagbar.

Während die Vorteile dieser Herangehensweise nicht zu bestreiten sind und sich schnell in anderen Unternehmen und Arbeitsbereichen durchsetzten, entwickelte sich ein neues Problem: Je nach Sichtweise stand nun nicht mehr das Projekt mit seinem Projektergebnis („*Die Mitarbeiter der Firma fertigen gemeinsam ein Produkt*") im Mittelpunkt, sondern vielmehr der Arbeitsprozess mit seinen verschiedenen Arbeitsphasen. Die Effizienz der Produktion war maßgeblich von der Qualität der Planungen der Arbeitsabläufe abhängig. Wurde ein Engpass nicht rechtzeitig erkannt, oder gingen Informationen und Teilergebnisse von einem Produktionsabschnitt zum nächsten verloren oder kamen verspätet hat, so hatte dies gravierende Konsequenzen auf die gesamte Produktion. Der Präsident von General Motors, Alfred P. Sloan jr. (1875-1966), erkannte dies und schuf ein erstes Managementsystem, auf dessen Basis Arbeitsabläufe überwacht und koordiniert werden konnten.

Parallel dazu wurden neue Unternehmensbereiche geschaffen, welche abgegrenzt vom reinen Produktionsbetrieb den Projektfortschritt analysierten, Planungsaufgaben durchführten und anhand vorliegender Daten den Arbeitsablauf steuern und geeignete Maßnahmen ergreifen konnten – woraus der Tätigkeitsbereich des Controllings maßgeblich hervorging. [Käm00]

Obwohl das „moderne Projektmanagement“ besonders auf militärischem Gebiet Mitte des 20. Jahrhunderts einen enormen Aufschwung erhielt, blieben die grundlegenden Ideen bis weit in die 80er Jahre hinein paradigmisch bestehen. Ein Wandel zu neueren Prozessmethoden wurde nötig, als man erkannte, dass sich das Marktumfeld zu stark veränderte und die strikte Trennung von Arbeitsaufgaben nicht länger haltbar war. Kollaboratives Arbeiten über Tätigkeitsgrenzen hinweg rückte stärker in den Mittelpunkt, und neue Managementmethoden wurden entwickelt, um vormals getrennte Aufgabenbereiche in einem kohärenten Unternehmensprozess modellieren zu können [Ham93]. Weiterhin schienen etablierte Managementstrukturen als immer häufiger aufgebläht und zu sehr auf Zahlen als auf Objekte der realen Welt fixiert. Eine Neuorganisation wurde nötig, welche sich stärker an Geschäftsprozessen orientierte und in der insbesondere der Begriff des Workflows eine zentrale Rolle einnahm.

Inzwischen findet man derartige moderne Organisationsformen nicht nur in großen Unternehmen. Während sich Projektplanung in früheren Jahren vorwiegend auf große, überaus komplexe Projekte bezog, findet man dies heutzutage in nahezu allen gesellschaftlichen Bereichen, ob nun im Geschäftsalltag von Großunternehmen, bei der Antragsabwicklung in kleinen und mittelständischen Unternehmen (KMUs), bei der Realisierung von Softwareprojekten im Software- Engineering oder auch im Privatsektor, wo durch eine Reihe von Organizer-Software besonders das Zeitmanagement im Mittelpunkt steht. Alle Anwendungen beweisen, dass Projektmanagement den effizienten Einsatz von Ressourcen ermöglicht, eine Rahmenstruktur für die Umsetzung eines Projektes vorgibt und ein optimales Ergebnis anstrebt, wobei ein Zugewinn im Mittelpunkt steht, der ohne entsprechende Nutzung womöglich nicht vorhanden wäre.

Aufgrund der Komplexität des Themas ist inzwischen eine unüberschaubare Anzahl an Büchern zum Thema Projektmanagement am Markt vorhanden. Vielfach werden im Alltag dabei die Begriffe Projektmanagement und Workflowmanagement oder Prozess und Workflow als Synonym gebraucht, obwohl im Kern wesentliche Unterschiede erkennbar sind. Vor der Analyse der Anforderungen an ein Projekt- und Workflowmanagementsystem soll daher zunächst eine klare Definition der Begriffe in diesem Zusammenhang erfolgen.

2.2. Begriffsdefinitionen

Begrifflichkeiten in Zusammenhang mit Projektmanagement werden im Alltag häufig mit unterschiedlichen inhaltlichen Deutungen angewendet. Die Abgrenzung ist meist unscharf – so kommt es vor, dass Konzepte mitunter widersprüchlich verwendet werden oder in einem falschen Kontext. Ursache dafür mag sein, dass sich in der vielfältigen Literatur zum Thema Projektmanagement unterschiedliche Vokabularien durch verschiedene Autoren und deren Standpunkt parallel entwickelt haben und dadurch in der Geschäftswelt eher flüchtig gebraucht werden. Die folgenden Begriffsdefinitionen sollen in erster Linie dazu dienen, einzelne Begriffe aus dem Gebiet des Projektmanagements klar voneinander abzugrenzen und zu Konzepten des realen Lebens Assoziationen herzustellen. Dies soll im nächsten Schritt dazu benutzt werden, die Begriffe hierarchisch zueinander in Beziehung zu setzen (Taxonomie) und darauf aufbauend in Kapitel 3 ein einfaches Modell (Ontologie) zu entwickeln, welches auf diesen Begriffen basiert und das zu realisierende System im Kern beschreibt.

2.2.1. Projekt

Der Brockhaus definiert den Begriff Projekt als „*ein geplantes oder bereits begonnenes, groß angelegtes Vorhaben*“. Die Ableitung vom lateinischen Begriff *projectum für etwas „nach vorn geworfenes“* suggeriert dabei die Zielstellung, mit dem Abschluss des Projektes eine Entwicklung vorangebracht zu haben, die es vorher in dieser Form nicht gab. Die ursprüngliche Bedeutung des Wortes Projekt bezog sich jedoch zunächst auf *etwas vorangegangenes* (lat. Pro – vorher), im Speziellen auf die Aufstellung eines Projektplanes, nicht auf die sich daran anschließende, konkrete Ausführung des Projektes. Die Bedeutung wechselte im anbrechenden Zeitalter der Industrialisierung, als ein Projekt nun als *die Gesamtheit aller Vorgänge aufgefasst wurde, die zum Erreichen einer vorgegebenen Zielstellung nötig sind*. Diese Definition ist dahingehend zu spezialisieren, dass verschiedene Disziplinen das Ziel eines Projektes unterschiedlich auffassen. So wird in wirtschaftswissenschaftlichen Publikationen ein Projekt als *ein betriebliches Vorhaben mit einem definierten Start- und Endtermin und einem klaren Auftrag* (vgl. QM-Lexikon unter www.qm-world.de) gesehen, *welches kosten- und abrechnungsmäßig separat erfasst werden kann*. Anderorts wird der Begriff Auftrag durch ein Kriterium ersetzt, dass ein Projekt *zu einer Veränderung der Strukturen* führen muss. Stärker technisch geprägte Definitionen sehen darin eher die *Realisierung einer Menge definierter Ergebnisse entsprechend vereinbarter Anforderungen* (Definition der International Project Management Association). In der DIN 69901 (Projekt-Management, 1987) wird der Begriff Projekt offiziell als *ein Vorhaben, das im wesentlichen durch eine Einmaligkeit der Bedingungen in ihrer Gesamtheit gekennzeichnet ist*, definiert. Aber auch diese Definition ist nicht in allen Fällen schlüssig genug, und wir unter anderem von Berner [Ben01] als „unsauber und nichtssagend“ abgetan.

In allen Definitionen finden sich dennoch Gemeinsamkeiten, die sich wiefolgt zusammenfassen lassen:

Definition 2: Projekt
Ein Projekt ist eine zeitlich begrenzte, längerfristige Unternehmung mit einmaligem Charakter, die unter einer wohl definierten, komplexen Aufgabenstellung ein neues Resultat erreichen möchte.

Ein Projekt wird damit über den Projektinhalt und Projektumfang definiert. Der Lösungsweg ist zu Beginn häufig unbekannt, nur eine Zielrichtung existiert, da aufgrund der Komplexität die gesamten Ausmaße eines Projektes mit allen Abhängigkeiten und gegensätzlichen Zielen nicht sofort zu erkennen sind. Ein Projekt zerfällt deshalb in der Regel in mehrere Phasen, die zusammen den Projektlebenszyklus bilden. Jede Phase besitzt eine entsprechende Zielsetzung, deren Erreichen den Abschluss einer Phase charakterisiert. Unterschiedliche Aufgaben im Projekt können dabei von verschiedenen Personen bearbeitet werden, sodass die kollaborative und interdisziplinäre Arbeit im Team eine zentrale Rolle einnimmt.

2.2.2. Phase

Eine Projektphase ist ein Möglichkeit, Projekte in zeitlich, logisch oder organisatorisch abgegrenzte Sinnabschnitte zu unterteilen. Eine Phase fasst eine Menge von Aktivitäten und Ereignissen zu einer Einheit zusammen. Am Ende jeder Phase steht eine Zielsetzung (oft als Meilenstein bezeichnet), die das zu realisierende Ergebnis zum Abschluss der Phase kennzeichnet. Ist dieses Ziel erreicht, so wird die Phase als abgeschlossen betrachtet. Phasen in einem Projekt müssen sich voneinander abgrenzen lassen und inhaltlich voneinander unterscheiden, das heißt, das Auftreten einer konkreten Phase im Projektverlauf muss eindeutig sein (einzelne Projektabschnitte können jedoch mehrfach (inkrementell) durchlaufen werden). Ziel der Aufteilung in Phasen ist es, das Gesamtprojekt besser planen und den Projektfortschritt besser überwachen zu können. In jeder Phase werden Prozesse festgelegt, welche zum Abschließen der Phase durchzuführen sind. Typische Phasen eines Projektes sind beispielsweise eine Analyse- oder Planungsphase, Implementierungsphase und Testphase. Die Abgrenzung von Phasen muss dabei nicht nur auf einer zeitlichen, sequentiellen Trennung beruhen, sondern kann sich auch auf anderen Kriterien stützen, zum Beispiel auf der Unterscheidung in unterschiedliche, zu realisierende Objekte.

Definition 3: Phase
Eine Phase ist die abstrakte Zusammenfassung einzelner Projektaktivitäten nach zeitlichen, logischen, oder organisatorischen Merkmalen zu voneinander trennbaren Projektabschnitten, an deren Ende ein konkretes Ergebnis vorliegt, welches im weiteren Projektverlauf weiterverwendet werden kann.

2.2.3. Prozess

Etymologisch stammt der Begriff aus dem Lateinischen (lat. processus = Bewegung, procedere = voranbringen) und kennzeichnet eine Abfolge von Tätigkeiten zur Erreichung eines Zieles oder zur Durchführung einer Aufgabe. Die jeweiligen Aktivitäten sind in der realen Welt durchführbar und hinsichtlich ihrer Durchführbarkeit und ihres Ergebnisses präzise beschreibbar. Modellhaft führt jede Ausführung einer Aktivität innerhalb eines Prozesses zu einem Zustandsübergang in dem beschriebenen System. Prozesse sind dabei zu einem beliebigen Zeitpunkt wiederholbar, stellen also feste, vordefinierte Handlungsanweisungen dar, was sie damit grundlegend von dem konzeptionellen Inhalt eines Projektes unterscheidet. Die Menge möglicher Ergebnisse eines Prozesses als logische Verknüpfung mehrerer Aktivitäten ist in der Regel deterministisch, kann andernfalls zumindest stochastisch beschrieben werden. Die Abarbeitung eines Prozesses wird durch Eintreten einer oder mehrerer Ereignisse ausgelöst (getriggert). Als Konsequenz daraus existiert ein Prozess nicht eigenständig, sondern muss einem konkreten Kontext zugeordnet sein, das heißt, es müssen bei den Eintreten des Ereignisses die zu bearbeitenden Objekte bekannt sein sowie eine Vorstellung vom Endprodukt nach Abschluss des Prozesses existieren. Im Gegensatz zu einem Projekt kann für einen Prozess häufig kein absolutes Start- und Enddatum angegeben werden. Vielmehr müssen in den übergeordneten Projektphasen Zeitspannen eingeplant werden, im Rahmen derer der konkrete Prozess zur Ausführung kommt. Ziel des Prozessmodells ist es, Planungsaufwand zu reduzieren, indem auf abstrakterer Ebene eine Folge fester, immer wiederkehrender Handlungsanweisungen zu einem Oberbegriff zusammengefasst werden kann. Jede Handlung in einem Prozess kann selbst wiederum einen Prozess darstellen, das heißt, Prozesse können schrittweise verfeinert werden. Ein elementarer Prozessschritt (Aktivität) ist erreicht, wenn eine weitere Unterteilung der Tätigkeit nicht möglich oder sinnvoll erscheint und die Tätigkeit direkt durchgeführt werden kann. Mehrere Prozesse können sowohl sequentiell als auch parallel von einem oder mehreren Beteiligten ausgeführt werden.

Definition 4: Prozess
Ein Prozess ist eine wiederholbare, im Kern eindeutig beschreibbare Tätigkeitsanweisung und kennzeichnet eine Abfolge von Arbeitsschritten zur Durchführung einer konkreten Aufgabe.

Besonderer Bedeutung kommt im Projektmanagement-Umfeld dem Begriff des Geschäftsprozesses zu, der von einem physikalischen oder technischen Prozess abzugrenzen ist und bei dem im Speziellen das Erreichen eines Geschäftsresultates im Vordergrund steht. Ein Geschäftsprozess ist eine Abfolge von Aktivitäten, die der Erbringung einer Dienstleistung dient. Er wird durch ein oder mehrere Ereignisse gestartet und durch ein oder mehrere bestimmte Ereignisse abgeschlossen. Die Besonderheit im Gegensatz zu anderen Prozesstypen liegt darin, dass unter dem Hintergrund des wirtschaftlichen Bezugs ein Geschäftsprozess nicht auf eine homogene Projektgruppe oder Betriebsabteilung beschränkt ist, sondern bewusst das Arbeiten über Abteilungsgrenzen hinweg betont.

So definiert Davenport 1993 einen Geschäftsprozess als „*a structured, measured set of activities designed to produce a specific output for a particular customer or market. It implies a strong emphasis on how work is done within an organization, in contrast to a product focus's emphasis on what. A process is thus a specific ordering of work activities across time and space, with a beginning and an end, and clearly defined inputs and outputs: a structure for action"* [Dav93].

Diese Strukturierung eines Geschäftsprozesses ist jedoch nicht mit einer real ausführbaren Arbeitsanweisung für eine beteiligte Person innerhalb dieses Prozesses zu verwechseln. Vielmehr liegt der Schwerpunkt auf einer konzeptionellen Trennung der verschiedenen Tätigkeiten innerhalb einer Organisation.

2.2.4. Workflow

Der Begriff Workflow und Prozess stehen zueinander in enger Beziehung. So wird ein Workflow im Brockhaus auch als die „*prozessorientierte Abwicklung arbeitsteiliger* Vorgänge […] *mit dem Ziel größtmöglicher Effizienz*" definiert, weswegen beide Begriffe häufig sogar in Synonymie verwendet werden. Vom informationstechnischen Standpunkt aus gesehen besteht jedoch dahingehend ein wesentlicher Unterschied, dass ein Workflow im engeren Sinne einer technischen Modellierung eines Prozesses entspricht, und damit eine direkt ausführbare Tätigkeitsanweisung darstellt. Das Fraunhofer Institut definiert den Begriff Workflow dahingehend, als dass er „*die an einem Arbeitsprozess beteiligten Personen und Arbeitsprozesse in einem Prozessmodell abbildet*" [Joh06]. Dem im Namen betonten „*Arbeitsablauf*" kommt dabei entscheidende Bedeutung zu, da innerhalb eines Workflows versucht wird, einzelne Arbeitsschritte (Aktivitäten) zu identifizieren und diese zueinander in Beziehung zu setzen. Sowohl die Arbeitsschritte als auch die Beziehungen sind hierbei eindeutig. Die Abfolge der Arbeitsschritte kann im Gegensatz zu einem Geschäftsprozess von einer beteiligten Person direkt umgesetzt werden, da diese von operativer Seite möglichst genau beschrieben wird. Rusinkiewicz beschreibt Workflows deshalb als „*Aktivitäten, welche die koordinierte Ausführung einer Reihe von Aufgaben durch unterschiedliche Verarbeitungseinheiten umfassen*" [Rus95]. Ziel eines Workflows ist es, den Arbeits- oder Informationsfluss mithilfe technischer Systeme so weit überwachen zu können, dass der Arbeitsverlauf für alle Beteiligten transparenter wird, die Durchführung qualitativ verbessert wird (indem keine vorab geplanten Tätigkeiten vergessen oder übergangen werden) und bestimmte Rahmenaufgaben automatisiert durchgeführt werden können und nicht mehr das Eingreifen eines Menschen erfordern, was wiederum einer Effizienzsteigerung des zugrunde liegenden Prozesses entspricht. Durch den engen Zusammenhang zwischen Workflows und Prozessen können Workflows ebenfalls beliebig verfeinert und wiederum in Workflows unterteilt werden, bis eine weitere Unterteilung nicht weiter möglich ist oder sinnvoll erscheint. Die beschriebene Tätigkeit ist dann elementar und kann eigenständig ausgeführt werden.

Definition 5: Workflow
Hinter einem Workflow verbirgt sich die informationstechnische Realisierung eines Prozesses in Form einer direkt ausführbaren Tätigkeitsanweisung, welche aus einzelnen durchzuführenden Aktivitäten besteht.

Workflows lassen sich auf unterschiedliche Art und Weise klassifizieren und beschreiben. Eine mögliche Klassifikation beruht auf der Beschreibbarkeit des auszuführenden Workflows. Neben gut strukturierten Workflows existieren dabei auch strukturlose Workflows, die nicht in eine Abfolge von Tätigkeiten zerlegt werden können. Sowohl bei elementaren Tätigkeiten als auch bei strukturlosen Workflows ist eine textuelle Beschreibung der Tätigkeit in irgendeiner Form nötig. Eine Übersicht über eine mögliche Kategorisierung des Workflow-Begriffs zeigt Tabelle 1.

1.	Stark strukturierte Workflows (Routine-Workflows) hohe Planbarkeit, relativ „geringe“ Komplexität, immer vorhersagbar
2.	Teilweise strukturierte Workflows (Regel-Workflows) sind noch kontrollierbar, Lösungsweg ist nur noch teilweise festgelegt
3.	Nicht strukturierte Workflows (ad-hoc-Workflows) nicht planbar und daher nicht vorhersagbar, Steuerungskomplexität wird zu groß

Tabelle 1: Kategorisierung von Workflows nach der Struktur

Daneben lassen sich Workflowmodelle in objektbasierte und aktivitätsbasierte Workflows einteilen. Objektbasierte (entity-based) Workflow beschreiben im Kern Dokumente, welche bis zu Ihrer endgültigen Fertigstellungen verschiedene Zustände durchlaufen. Im Mittelpunkt der Beschreibung steht der einzelne Arbeitsschritt („*work item*“), wie das entsprechende Objekt in dem jeweiligen Zustand verändert wird. Dem gegenüber stehen aktivitätsbasierte Workflows, welche eine Abfolge durchzuführender Tätigkeiten in den Mittelpunkt stellen, um die Gesamtaufgabe abzuschließen. Beide Konzepte werden kontrovers diskutiert, wobei die Meinung vertreten wird, dass objektbasierte Workflows für einfache Sachverhalte sehr geeignet sind, sehr komplexe Tätigkeiten aber nur mit einem aktivitätsbasierten Workflowkonzept realisiert werden könnten. Parallel dazu gibt es Meinungen, dass im Grunde beide Konzepte auf einer niedrigeren Ebene ineinander überführbar sind[4].

[4] vgl. http://wiki.zope.org/zope3/TryingToUnifiyWorkflowConcepts

Als letztes Klassifikationsmerkmal können Workflows in zustandsbasierte und nachrichtenorientierte Workflows eingeteilt werden, je nachdem, nach welchem Prinzip die Abarbeitung eines Workflows geschieht. Zustandsbasierte Workflows sehen einen Workflow als Abfolge einzelner Aktivitäten an, zwischen welchen Übergänge existieren, wobei der aktuelle Ausführungszustand durch eine Workflow-Engine verwaltet wird. Im Gegensatz dazu existieren ebenso Workflowausführungsmodelle, welche auf das Eintreffen von Nachrichten reagieren, mit anderen Systemen ständig in Kontakt stehen, und der Workflow basierend auf eintreffenden Ereignissen voranschreitet.[5]

2.2.5. Aktivität

In den vergangenen Definitionen wurde bereits mehrmals der Begriff Aktivität verwendet. Dieser Begriff kann wiefolgt aufgefasst werden:

Definition 6: Aktivität
Eine Aktivität ist eine elementare Tätigkeit, welche nur noch deskriptiv beschrieben werden kann und direkt ausführbar ist und damit den kleinsten Baustein in einem Arbeitsablauf bildet.

Beschrieben werden kann eine Aktivität innerhalb eines Workflows durch den Zeitpunkt der Ausführung, benötigte Ressourcen zur Durchführung, einer Beschreibung der Tätigkeit mit einer eventuellen Spezifikation zu benutzender, externer Systeme und durch eine Auflistung der an der Ausführung beteiligten Personen. In manchen Publikationen wird statt des Begriffs „Aktivität“ eher der Begriff Aktion verwendet mit dem Verweis darauf, dass bei Verwendung der Bezeichnung Aktivität nicht eindeutig herausgestellt wäre, ob man in einem Workflow damit den Zustand oder den Zustandsübergang meint. Im Folgenden sei der Begriff Aktivität mit dem Wort Tätigkeit oder Arbeitsschritt gleichgesetzt und beziehe sich auf eine jeweilige auszuführende Aufgabe in einem konkreten Zustand des Workflows.

2.2.6. Projektmanagement

Zur Einordnung des Begriffes Projektmanagement diene erneut eine lexikalische Definition aus dem Brockhaus, der Projektmanagement definiert al die *Gesamtheit der Planungs-, Steuerungs- und Kontrollaktivitäten, die bei relativ innovativen und risikobehafteten Vorhaben mit komplexer Struktur, vorgegebenen Terminen und limitierten Kosten anfallen, sowie für die fachbereichsübergreifende Koordination dieser Führungstätigkeiten.* Im Wesentlichen umfasst diese Definition Methoden zur Durchführung eines Projektes. Im PMBOK Guide des Projektmanagement-Instituts wird dies weiter ausgeführt als „*the application of knowledge, skills, tools and techniques to project activities to meet project requirements*” [PMI04].

[5] vgl. http://www.jboss.com/products/jbpm/stateofworkflow

Projektmanagement ist damit eine Tätigkeit, welche sowohl konzeptionelle Aufgaben wie das Identifizieren von Projektzielen und dem Abschätzen der Projektkomplexität, eine Beschreibung des Projektverlaufs sowie entsprechenden Ressourcenbedürfnissen, als auch die Überwachung der Projektdurchführung, eventuelle Umplanung, das Erreichen des Projektergebnisses, die Qualitätssicherung und auch eine Nachbetrachtung des durchgeführten Projektes umfasst. Daneben spielen weitere Aufgaben aus anderen Disziplinen – beispielsweise die Führung und Motivation eines Projektteams – eine wesentliche Rolle. Ziel ist es, durch den Einsatz von Projektmanagement die Fertigstellung eines Projektes unter begrenzten Ressourcen (Rahmenbedingungen bezüglich Zeit, Geld, Personal, Qualität u.a.) sicherzustellen. Für weitere Betrachtungen aus wirtschaftlicher Sicht sei auf weiterführende Literatur verwiesen; im Folgenden stehe vor allem die softwarebasierte Unterstützung von Projektmanagementaufgaben durch Projekt-Management-Systeme im Mittelpunkt. Diese sind in der Regel auf wohl definierte Anwendungsdomains spezialisiert und reichen in ihrer Komplexität von einfachen, kostenfreien Editoren bis hin zu komplexen und preisintensiven kollaborativen Softwaresystemen. Universelle Projekt-Management-Systeme, welche die Abwicklung von Projekten aus beliebigen Anwendungsbereichen erlauben, werden oftmals in Ihrer Einsetzbarkeit in Frage gestellt („*One size doesn't fit all*"). Vielmehr ist die Wahl eines geeigneten Projektmanagementtools von entscheidender Bedeutung – zu kleine Systeme genügen nicht den gestellten Anforderungen, zu ausgereifte Systeme können dem gegenüber zu einer zu komplizierten Bedienung für den konkreten Anwendungsfall führen. Im Allgemeinen bieten alle Projektmanagement-Systeme gewisse Funktionalitäten zur Verwaltung der Aufgaben von Projektbeteiligen, zur Projektplanung, der Dokumentenverwaltung und der Projektsteuerung. Gantt-Diagramme, Aufgabenlisten, Fortschrittsüberwachung, Terminplaner, Kontrollmechanismen, Kostenverwaltungsmodule und Reportfunktionen gehören zu typischen Funktionalitäten, welche durch Projektmanagement-Systeme geboten werden. In Kapitel 2.5 und 2.6 werden diese Softwaretools genauer untersucht.

2.2.7. Prozessmanagement

Während sich Projektmanagement auf die Umsetzung von Projekten beschränkt, umfasst Prozessmanagement die Überwachung von Prozessverläufen. Im Mittelpunkt steht die kosteneffektive Durchführung von Prozessen als Abfolge verschiedener Aktivitäten. Dazu ist eine genaue Beschreibung, Darstellung und Überwachung des Ablaufs einer Aktivität nötig, welcher durch Einsatz von Prozessmanagementmethoden stetig optimiert werden kann. Ein Prozess ist dabei im engeren Sinne als Geschäftsprozess zu sehen, das heißt, es stehen weniger sequentielle Arbeitsanweisungen im Mittelpunkt als vielmehr bereichsübergreifende Aufgaben, welche ausgeführt werden, um in ihrer Gesamtheit ein gemeinsames Ergebnis zu erreichen. Um den Fluss zwischen den einzelnen Instanzen zu verbessern und damit den Prozess zu optimieren, sind zum einen eine genaue Planung der Geschäftsprozesse, als auch die Verwendung vergleichbarer Messgrößen nötig.

2.2.8. Workflowmanagement

Im Mittelpunkt des Workflowmanagements steht die Unterstützung von Arbeitsabläufen durch technische Systeme (Workflow-Management-Systeme) mit dem Ziel der Effizienzsteigerung. Dies beinhaltet die Modellierung, Ausführung und Überwachung von Arbeitsprozessen in Form von Workflows. Modellierung umfasst hierbei die Darstellung von Teilen eines Geschäftsprozesses in Form von sequentiellen oder parallelen Arbeitsschritten, Ausführung die computergestützte Durchführung dieser Arbeitsschritte (möglicherweise durch Integration anderer Softwaresysteme), sowie Überwachung in erster Linie die Sicherstellung des Arbeitsfortschrittes durch Bereitstellung von Informationen über den bisherigen Prozessverlauf. Im Prinzip ist ein ähnliches Herangehen auch bei Projektmanagementsystemen, allerdings auf einer abstrakteren Ebene, wiedererkennbar. Basierend auf den vorgestellten Definitionen von Projekt und Prozess und der hierarchischen Beziehung zwischen diesen beiden Begriffen wäre es vorstellbar, ein Workflowmanagement als Teil eines Projektmanagementsystems aufzufassen. Dies war in der Praxis bis vor kurzem jedoch eher die Ausnahme. Eine Vielzahl von Workflowmanagementsystemen wurde zur operationalen Durchführung prozessorientierter Abläufe genutzt, während Projektmanagementsysteme vor allem zur Projektplanung genutzt worden. Die Vorteile der Kombination beider Systeme wurden unter anderem durch Bauer [Bau04] analysiert. Darin wird davon ausgegangen, dass Projektmanagementsysteme und Workflowmanagementsysteme teils vollkommen unterschiedliche Datenrepräsentationen verwenden und gleichzeitig unterschiedliche Funktionen bereitstellen, sodass eine direkte Systemintegration häufig nicht möglich sei. Sinnvoll wäre dies aber, da darüber Projektplanungsdaten für die Prozessbearbeitung bereitgestellt werden könnten, andererseits aber auch der aktuelle Bearbeitungszustand für die Projektplanung verfügbar wäre. Um dies zu erreichen, wird ein möglicher Weg in der Einführung einer Zwischenschicht gesehen, welche die Aggregation und Verteilung der Daten übernimmt.

2.2.9. Groupware

Im Zusammenhang mit Projektmanagement und kollaborativen Systemen wird häufig der Begriff Groupware genannt, welcher schlicht mit „Gruppenprogramme“ übersetzt werden könnte. Statt einer Reihe autarker Programme verbirgt sich daher aber häufig ein System zur Unterstützung kollaborativen Arbeitens, welches die Koordination von Aufgaben und Terminen sowie den systematischen Austausch von Dokumenten im Team ermöglicht. Der Fokus von Groupware liegt in der Bereitstellung von Funktionalitäten, welche die gemeinsame Arbeit an einer Ressource (bspw. mittels Multi-User-Editoren) ermöglichen und dafür vielfältige Interaktionsmöglichkeiten bieten. Im Gegensatz dazu wird in Workflow-Management-Systemen vor allem die Prozessstruktur und der Arbeitsfortschritt betont, während Groupwaresysteme häufig auf beliebigen, unstrukturierten Daten arbeiten können.

2.2.10. Knowledgemanagement

Ein letzter Aspekt, welcher bei der Verwaltung von Prozessen in Unternehmen und Institutionen eine wesentliche Rolle spielt, findet sich im Wissensmanagement (Knowledge Management). Dies ist darin begründet, dass die Durchführung von Arbeitsvorgängen möglichst gut dokumentiert werden muss, um eine erneute Ausführung zu einem späteren Zeitpunkt zu ermöglichen, ohne dass bisher gewonnene Erkenntnisse verloren gehen oder erneut erworben werden müssen (was unmittelbar zu einer Effizienzverschlechterung führt). Wissensmanagement umfasst jedoch nicht nur die Sicherung und den Einsatz vorhandenen Wissens, sondern ebenso die Generierung und Dokumentation neuen Wissens, welches in eine Wissensbasis einfließt, auf die immer wieder zurückgegriffen werden kann. Wissen ist ein relativ unscharfer Begriff, welcher in entsprechender Literatur zum Thema Knowledge Management unterschiedlich gedeutet wird. Im Folgenden umfasst die Wissensbasis neben der Beschreibung von Arbeitsabläufen auch alle damit verbundenen Daten, auf welche im Idealfall jeder Mitarbeiter zugreifen und darauf aufbauend die Wissensbasis erweitern kann. Dieser Ansatz ist für die Entwicklung eines Projekt- und Workflowmanagementsystems in jedem Fall zu berücksichtigen, wenn einzelne Aktivitäten definiert, beschrieben und in verschiedenen Prozessen verwendet werden.

2.3. Modellierungsansätze

Nachdem in 2.2. grundlegende Begriffe geklärt worden, ist zu einer Umsetzung in einem realen System zunächst eine geeignete Notations- und Darstellungsform zu suchen. Verschiedene Ansätze zur Darstellung von Workflows wurden dazu in den vergangenen Jahren entwickelt. Diese seien in diesem Kapitel näher vorgestellt. Im Gegensatz dazu finden sich im Projektmanagementbereich kaum standardisierte grafische Notationen oder Austauschformate. Im Hinblick auf die Entwicklung eines semantikbasierten Projektmanagementsystems soll dieser Abschnitt daher nach einer geeigneten Notation suchen, welche auch im Projektmanagementbereich sinnvoll nutzbar sei.

Die Durchführung verschiedener Aufgaben kann durch eine Aufschlüsselung in einzelne Arbeitsschritte (Aktivitäten) beschrieben werden, welche zeitlich zueinander in Beziehung gesetzt sind (sequentielle, parallele oder alternative Abarbeitungsmöglichkeiten). Die Modellierung dieser Aufgabenliste als Workflow basiert in der Regel auf einer Umsetzung als gerichteter, endlicher Graph mit einem (oder mehreren) ausgezeichneten Start- und Endzuständen. Dabei erfolgt modellbedingt eine Abstraktion verschieden starken Ausmaßes, da Tätigkeiten der realen Welt nur bis zu einem gewissen Grad strukturell dargestellt werden können. Im Mittelpunkt der Modellierung steht die Darstellung der Prozessausführung, die technisch bestmöglich unterstützt wird. Daneben spielen aber auch Daten über die modellierten Prozesse eine wesentliche Rolle.

Dies führt zu einer semiformalen Darstellung des zugrunde liegenden Arbeitsablaufs, da die Beziehung zwischen einzelnen Arbeitsschritten visuell abgebildet wird (das bedeutet, für einen Menschen einfach zu verstehen), die Graphstruktur selbst auch algorithmisch abbildbar ist, zusätzliche Informationen über die einzelnen Arbeitsvorgänge aber häufig den entsprechenden Knoten informell (das heißt, als textuelle Beschreibung) zugeordnet werden. Für eine computergeschützte Überwachung einer Tätigkeit ist eine rein formale Beschreibung der Tätigkeiten geeigneter, welche jedoch häufig aufwändiger zu formulieren und für einen Menschen schwieriger zu verstehen ist. Als Kompromiss verwenden aktuelle Workflow-Managementsysteme semiformale Beschreibungen, deren Vertreter Gadatsch entsprechend Abbildung 1 in die Kategorien Objektorientiert, Datenflussorientiert und Kontrollflussorientiert einordnet [Gad03]. Zoll [Zol06] erweitert dieses Schema um regelbasierte Methoden. In der Kategorisierung finden sich sowohl das klassische Modell der Ereignisgesteuerten Prozessketten (EPKs) als auch Modellierungsansätze basierend auf UML wieder. Darüber hinaus existiert seit 2002 neben UML eine weitere, zunehmend häufig benutzte Notationssprache in Form der Business Process Modelling Notation (BPMN).

Alle Modellierungsansätze haben gemeinsam, dass sie von einem Grundvokabular mit elementaren Bausteinen ausgehen (welches sich an den vorgestellten Definitionen aus Abschnitt 2.2 orientiert), aus denen komplexere Workflows aufgebaut sind. Den elementaren Bausteinen können Metainformationen wie etwa zu Beginn benötigte Ressourcen, Ausgabeobjekte und Transformationsregeln zugeordnet werden. Primäres Ziel ist es, Funktionsabfolgen für den Betrachter übersichtlich darzustellen. Darüber hinaus muss das Modell so formal aufgebaut sein, dass eine Teilautomatisierung durch andere Software ermöglicht wird. Im Folgenden sind wesentliche Modellierungsmethoden genauer vorgestellt.

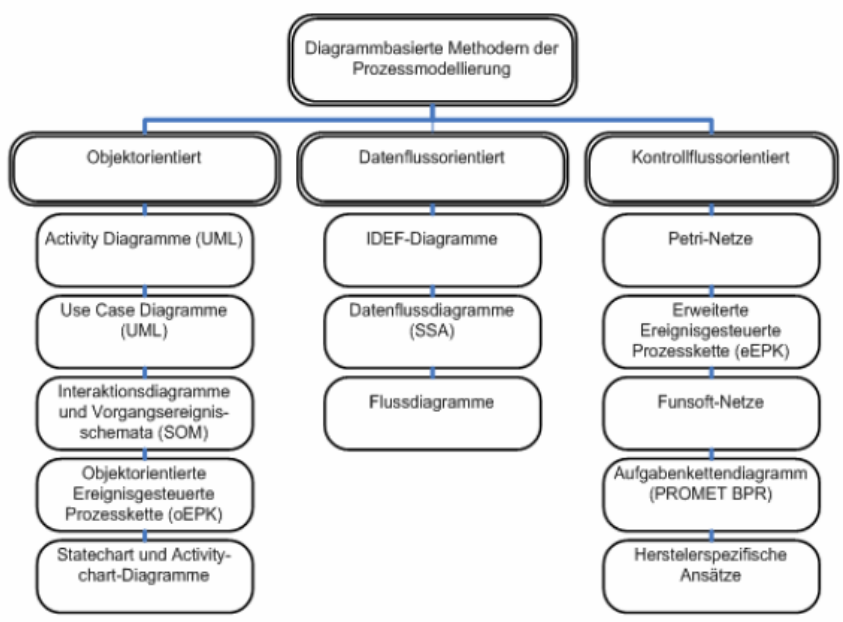


Abbildung 3: Methoden der Prozessmodellierung nach Gadatsch [Gad03]

2.3.1. Ereignisgesteuerte Prozessketten

Ereignisgesteuerte Prozessketten (EPKs) bieten ein Modell zur Darstellung von Arbeitsabläufen, welches 1992 in Deutschland an der Universität des Saarlandes entwickelt wurde und seitdem vielfach eingesetzt wird.[6] Grundidee hinter EPKs ist die Unterteilung von einzelnen Arbeitsschritten in auszuführende Tätigkeiten (Funktionen) und Ereignisse, welche die Ausführung der Folgetätigkeiten auslösen und mehrere Funktionen miteinander verbinden. Eine EPK ist damit ein gerichteter, bipartiter Graph aus Ereignissen und Funktionen. Daneben sind Informationsobjekte Bestandteil des Prozessmodells, auf welche durch Funktionen zurückgegriffen werden kann. Weiterhin finden sich logische Operator-Bausteine zur Ausführung einer Tätigkeit in Abhängigkeit mehrerer, vorausgegangener Funktionen, sowie Bindeglieder, welche neben einer sequentiellen Durchführung auch eine parallele oder alternative Ausführung von Funktionen im Prozessmodell ermöglichen. Erweiterte EPKs (eEPKs) können darüber hinaus weitere zusätzliche Objekte enthalten. Mithilfe der XML-basierten Event-driven Process Chain Markup Language (EPML) wird ein plattformübergreifendes Austauschformat bereitgestellt. Wesentlicher Vorteil von EPKs ist die einfache Handhabung. Kritiker sehen dem gegenüber in EPKs ein wesentliches Problem, dass deren Syntax und Semantik formal nicht ausreichend definiert ist und daher in der Praxis beliebige und vor allem mehrdeutige Strukturen auftreten können. Dies erschwere den Austausch und die Analyse in Workflow-Management-Systemen. So zeigt beispielsweise van der Aalst auf, dass Ereignisgesteuerte Prozessketten als Untermenge von Petri-Netzen aufgefasst und damit in diese überführt werden können und dies eine Reihe von Vorteilen bringt [Aal98]. Darüber hinaus beschäftigt sich Kindler mit dem Problem, nicht-lokale Semantiken für EPKs zu definieren, indem er dazu ein mathematisches Grundgerüst schafft [Kin03].

2.3.2. Flussdiagramme

Ein Flussdiagramm ist eine grafische Darstellungsmöglichkeit für beliebige Arbeitsabläufe. Es fand vor einigen Jahren besondere Anwendung bei der Darstellung von Programmabläufen (Algorithmen), ist prinzipiell aber zur Darstellung beliebiger Prozesse geeignet. Im Mittelpunkt steht eine sequentielle Abarbeitung verschiedener Tätigkeiten, um von einem Startzustand in einen Endzustand zu gelangen. Neben der Möglichkeit, einzelne Operationen sowie Ein- und Ausgabewerte zu definieren ermöglicht ein Verzweigungsoperator, Alternativen während der Abarbeitung zu berücksichtigen, die in der Regel binär beantwortbar sind. Weiterhin existiert die Möglichkeit, Sprungstellen in andere Flussdiagramme zu definieren.

[6] So wurden EPKs lange Zeit im SAP R/3-Systemen zur Prozessdokumentation verwendet.

Alle gültigen Symbole eines Flussdiagramms sind in der DIN 66001 genormt. In der Gesamtbewertung bieten Flussdiagramme zur Darstellung von Workflows jedoch nicht genügend Möglichkeiten, da parallele Ausführungen als auch die objektorientierte Modellierung nicht ausreichend unterstützt werden, was nicht zuletzt in dem Entwicklungszeitpunkt Ende der Vierziger Jahre begründet ist. Dennoch existieren genügend Werkzeuge zur Erstellung von Flussdiagrammen. Im Mittelpunkt steht dabei die grafische Darstellung des Prozesses. Daraus resultierend haben sich im Gegensatz zu anderen Modellierungsmethoden für Prozesse keine Beschreibungssprachen zum Austausch und zur automatischen Verarbeitung von Flussdiagrammen durchgesetzt.

2.3.3. Petri Netze

Petri-Netze wurden 1968 durch Carl Adam Petri entwickelt und sollen es ermöglichen, beliebige Prozesse formal zu beschreiben. Anstatt sich auf einen sequentiellen Kontrollfluss zu beschränken, erweitern Petri-Netze das klassische Ausführungsmodell dahingehend, dass mit Ihnen nebenläufige Anweisungen in verteilten Systemumgebungen problemlos modelliert werden können. Die aus der klassischen Automatentheorie bekannten Knoten werden dazu um eine Markierung (token) erweitert, welche generiert oder entfernt werden kann. Die von diesen Knoten ausgehenden gerichteten Kanten treffen sich in einem (oder mehreren) Übergangsobjekt(en) (transition), welches erst durchlaufen werden kann, wenn genügend Markierungen vorhanden sind. Übertragen auf das Konzept eines Workflows kann dies dahingehend gedeutet werden, dass erst eine Reihe vorausgehender paralleler Aktivitäten durchgeführt worden sein muss, ehe der Gesamtprozess ab einem bestimmten Punkt weiter fortschreiten kann. Mit diesen Mitteln sind Petri-Netze einfach und ausdrucksstark, aber gleichzeitig formal genug, um damit Geschäftsprozesse in Workflow-Management-Systemen modellieren zu können. Ähnlich wie bei EPKs existieren zum Austausch und zur Analyse von Petri-Netzen eine Reihe von Werkzeugen und Dokumentenformaten. Interessant ist dabei neben Varianten der Graphical XML Schema Definition Language (GXSL) vor allem die (XML-basierte) Petri Net Markup Language (PNML), welche die Modellierung höherer (dann als XML-Netze bezeichneter) Petri-Netze ermöglicht. Daneben existiert als Open-Source Projekt eine Workflow-Beschreibungssprache unter der Bezeichnung Yet Another Workflow Language (YAWL), welche in erster Linie eine bestmögliche Umsetzung gängiger Workflow Patterns verfolgt und dazu auf Petri-Netzen aufsetzt.

2.3.4. Unified Modeling Language

Die Unified Modeling Language (UML) ist eine 1997 von der Object Management Group standardisierte Sprache zur (objektorientierten) Beschreibung von Softwaresystemen, die bis 2005 zu UML2 erweitert wurde. Durch die Unified Modeling Language werden in erster Linie Begriffe zur Modellierung beliebiger Anwendungssysteme sowie deren Beziehung untereinander spezifiziert. Darauf aufbauend enthält die UML in insgesamt 13 Diagrammtypen sowohl Möglichkeiten zur graphischen Darstellung verschiedenster Modelle, als auch ein auf XML basierendes Austauschformat dieser Modelle (UML 2.0 Diagram Interchange, hervorgegangen aus dem XML Metadata Interchange Format (XMI)). Mithilfe des universellen Modellierungsansatzes und einem Austauschformat, welches sowohl die Weitergabe von physischen Diagramminformationen als auch von Semantik ermöglicht, war die Basis für einen plattformübergreifenden Einsatz geschaffen. Ein wesentlicher Punkt dabei war die Standardisierung wesentlicher Begriffe zur Modellierung von Anwendungssystemen. Das Vokabular ist dabei um einiges umfangreicher, als dies etwa bei EPKs der Fall ist, und hierarchisch in als Spracheinheiten (language units) bezeichneten Gruppen zusammengefasst, die wiederum in verschiedenen Modellierungsszenarien innerhalb der 13 möglichen Diagrammtypen von UML zur Anwendung kommen. Zur Verhaltensmodellierung spielen dabei die Spracheinheiten Aktivität und Aktion eine übergeordnete Rolle, wobei unter Aktion eine elementare Tätigkeit verstanden wird, mit denen einzelne Aktivitäten modelliert werden, was wiederum in einem Aktivitätsdiagramm dargestellt werden kann.

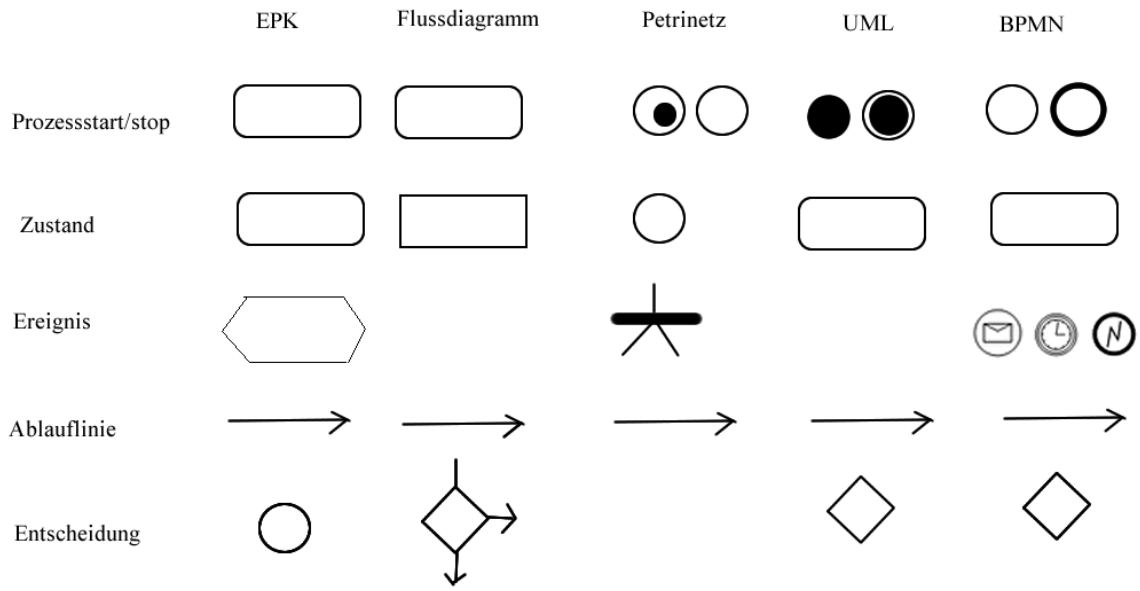


Abbildung 4: Vergleich grundlegender Elemente verschiedener Modellierungsansätze

2.3.5. Business Process Modeling Notation

Die Business Process Modeling Notation (BPMN) ist eine vergleichsweise junge (seit 2002 entwickelte) Möglichkeit zur Modellierung und Darstellung von Geschäftsprozessen, welche von einem Non-Profit-Konsortium (BPMI) als offener Standard entwickelt wurde. Sie ist seit 2006 neben UML ein weiterer Standard der Object Management Group. Im Mittelpunkt steht die graphische Darstellung von Arbeitsabläufen, sodass diese für den Menschen einfach überblickbar und verständlich sind. Um eine darüber hinausgehende computergestützte Verarbeitung zu ermöglichen, definiert die BPMN eine Transformation, um fertige Business Process Diagramme in ein XML-basiertes Format zu überführen. Von besonderer Bedeutung sind dabei die Business Process Execution Language for Web Services (WS-BPEL) sowie die XML Process Definition Language (XPDL). Ähnlich wie andere, bereits beschriebene Modellierungsansätze verwendet die BPMN eine Reihe vordefinierter Bausteine, mit denen komplexe Geschäftsprozesse modelliert werden können. Ausgehend von dem Aktivitätsbegriff wird besonders eine hierarchische Zerlegung in Subaktivitäten sowie eine Trennung verschiedener Benutzerrollen durch so genannte Pools und Swimlanes betont.

2.3.6. Zusammenfassung

Die Vorstellung der verschiedenen Modellierungsmethoden für Geschäftsprozesse hat gezeigt, dass seit Beginn der Neunziger Jahre zwar mehrere, konzeptionell verschiedene Ansätze existieren, diese sich bei der Anwendung in einem realen Workflow-Management-System aber nur punktuell in der Bezeichnung der zur Verfügung stehenden Modellbausteine und deren Ausdruckskraft unterscheiden. Wesentliche Elemente wie Aktivitäten (elementare Aktionen), Entscheidungsobjekte, Ereignisse und Objekte zur Aggregation oder Aufsplittung mehrerer Workflows finden sich in allen Konzepten in unterschiedlicher Ausprägung wieder (Abbildung 4). Aus diesem Grund mag es ausreichen, sich auf eine Begrifflichkeit bei dem Entwurf eines neuen Workflow- oder Projektmanagementsystems festzulegen, ohne dass dadurch ein proprietäres Modell geschaffen wird, welches zu anderen Begriffsdomänen inkompatibel ist. Interessant ist weiterhin, dass sich in der Regel zu jedem der Modelle ein entsprechendes, XML-basiertes Austauschformat entwickelt hat. Bei den Formaten, welche anfänglich durch unterschiedliche Firmen und Initiativen entwickelt wurden, zeichnet sich zunehmend ab, dass diese zueinander kompatibel und in offenen Standards zusammengefasst werden, wie Abbildung 5 illustriert.

Ein weiteres Problem, welches im Zusammenhang mit jeder der Modellierungsmethoden genannt wird, ist die Darstellung und Übertragung von semantischen Informationen. Während der Syntax eines Modells relativ einfach definierbar und validierbar ist, ist dies auf semantischer Ebene bisher nicht ideal gelungen. Die Fragestellung, was der Begriff Semantik bei der Modellierung von Geschäftsprozessen beinhaltet, wird in Kapitel 3 erneut aufgegriffen und vertieft.

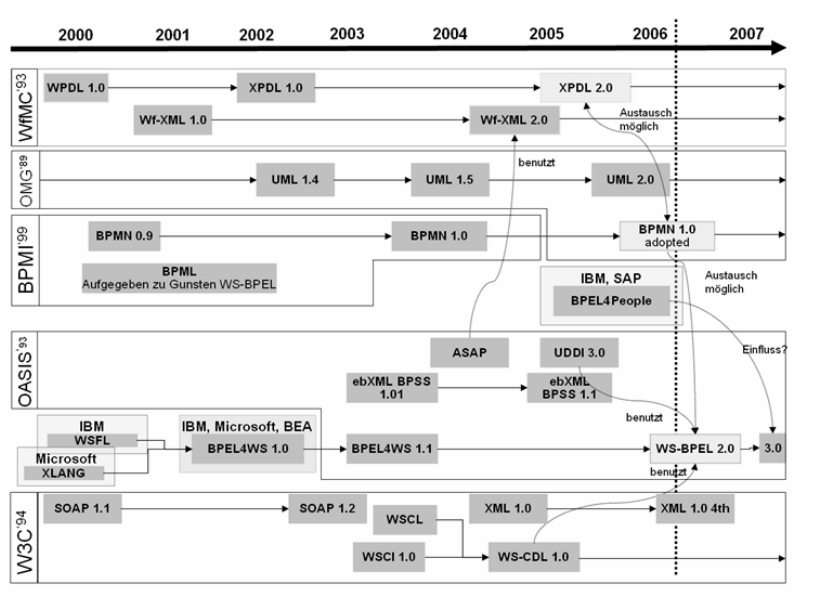


Abbildung 5: Entwicklung von Workflowbeschreibungsformaten, Quelle: http://www.citt-online.com/downloads/exp4/Bartonitz_CITT_Regensburg_20061212.ppt#284,4,Warum Standards für BPM/Workflow?

2.4. Anwendungsszenarien

2.4.1. Motivation

Um im Folgenden bestehende Systeme testen und miteinander vergleichen zu können, sollen nachfolgend fünf typische Szenarien entworfen werden, welche mit existierenden Applikationen umsetzbar sein sollten. Die Anwendungsszenarien sind dabei so gewählt, dass sowohl die Aspekte Projekt, Prozess als auch das kollaborative Arbeiten betont werden. Die Modell-Komplexität nimmt in der Reihenfolge der Beispiele zu.

2.4.2. Spaghetti kochen

Erstes einfaches Beispiel sei eine Tätigkeitsanweisung aus dem Alltag, um Spaghetti (beispielsweise alla Bolognese) zuzubereiten. Da der Ablauf überschaubar und weitestgehend vorhersagbar ist, als Einpersonenaktivität angesehen werden kann und das Ende der Ausführung nicht im Voraus auf einen festgesetzten Zeitpunkt terminiert werden kann, handelt es sich in diesem Fall nach der Definition nicht um ein Projekt, sondern vielmehr um einen Prozess, der aus einzelnen Aktivitäten besteht. Der Prozess wiederum lässt sich zerlegen in zwei Sub-Prozesse *Spaghetti kochen* und *Soße zubereiten*. *Spaghetti kochen* umfasst dabei die Tätigkeiten, einen Topf mit Wasser aufzusetzen, Salz hinzu zu geben, wenn das Wasser kocht, die Spaghetti hinzuzufügen und nachdem diese weich genug sind, das Wasser aus dem Topf abzugießen. Parallel dazu kann der zweite Subprozess durchgeführt werden, der darin besteht, Margarine in einer Pfanne zu erhitzen, Fleisch anzubraten, das Gemüse zu schneiden, schließlich Tomatenmark hinzuzufügen und das Ganze zu salzen und pfeffern (dabei könnte dieser Prozess beliebig weiter unterteilt werden).

2.4.3. Abendessen mit Freunden

Der in 2.4.2 dargestellte Workflow kann beispielsweise in einem komplexeren Projekt als Einzelelement benutzt werden. Als entsprechendes Szenario diene im Folgenden die Situation, ein *Abendessen mit Freunden* zu organisieren. Da dies in der Regel die Interaktion mehrerer Personen umfasst, auf einen bestimmten Termin festgelegt und in mehrere Phasen unterteilbar ist, deren Ausgang nicht immer vorhersagbar ist, kann dieses Szenario durchaus als kleines Projekt aufgefasst werden. So ist es in einer ersten Phase nötig, die entsprechenden Gäste einzuladen und auf deren Teilnahmebestätigung zu warten. Während die Gäste in der Regel ein Gastgeschenk besorgen, hat der Gastgeber die Aufgabe, die nötigen Zutaten für das Abendessen einkaufen zu gehen. Eine zweite Phase enthält daran anschließend die Zubereitung des Abendessens (siehe 2.4.2) als auch den Empfang der Gäste, das Servieren und Essen, als auch das Verabschieden der Gäste. Als weitere Phase kann die Nachbereitung des Abends angesehen werden, worin vor allem Aufräumaufgaben und das Einholen eines Feedbacks als wesentliche Tätigkeiten zu finden sind.

2.4.4. Diplomarbeit schreiben

Während die Zuhilfenahme eines Projektmanagementwerkzeugs in den zwei vorausgegangenen Szenarien eher fragwürdig und hypothetisch zu sehen ist, soll nun ein praktischerer Anwendungsfall betrachtet werden, der in der *Erstellung einer wissenschaftlichen Arbeit* besteht. Während dies je nach Blickwinkel bei einfachen Ausarbeitungen durchaus als Prozess aufzufassen sein kann (da Akademiker während ihrer Laufbahn nach dem gleichen Schema immer wiederkehrend Zusammenfassungen ihrer eigenen Forschungsarbeit schreiben), ist dies für einen Studierenden bei der Anfertigung seiner Diplomarbeit nicht der Fall. Zwar sind ein Start- und Abgabetermin bekannt und ebenso eine grobe Festsetzung des Themas, welche jedoch in der Gesamtheit zu Beginn häufig nicht vollständig verstanden werden kann und sich entwickelt, wodurch ein Projekt gekennzeichnet ist. Auch hier zerfällt der weitere Projektverlauf in verschiedene Phasen. So ist der Abschluss der ersten Phase durch die Festlegung des Themas und des Starttermins der Diplomarbeit gekennzeichnet, was wiederum eine Kontaktaufnahme mit einem oder mehreren Betreuern, sowie mehrere Treffen und das Fixieren des Themas in einem Dokument einschließt. Dieses Thema muss nicht nur der Betreuer bestätigen, sondern auch durch den zuständigen Prüfungsausschuss genehmigt und im entsprechenden Prüfungsamt verarbeitet werden. In einer darauf folgenden Einarbeitungsphase steht vor allem eine gründliche Literaturrecherche im Mittelpunkt. Darüber hinaus sind wesentliche Begriffe als auch Rahmenbedingungen mit dem oder den zuständigen Betreuern zu klären. Ergebnis dieser Phase könnte eine Art Visionsdokument sein, in dem die konkrete Herangehensweise an die Bearbeitung des Themas dargelegt wird. Je nach Themengebiet beinhaltet eine dritte Phase die praktische Umsetzung (Implementierung) der gewählten Aufgabe. In der IT umfasst dies beispielsweise eine Machbarkeitsstudie, Spezifikation, Programmierung und den Test des zu entwickelnden Systems.

Die Ergebnisse daraus sind schriftlich zu fixieren, wodurch mehrere, in ihrer Vollständigkeit zunehmende Versionen des Diplomarbeit-Dokuments entstehen. Diese wiederum sind mehrfach zu prüfen, zu modifizieren und vor Abgabe mehrfach korrekturzulesen. Das Ende dieser Phase wird durch die Abgabe der fertigen Diplomarbeit gekennzeichnet. In einer letzten Phase ist die Arbeit durch die Betreuer zu bewerten sowie durch den Studierenden eine Präsentation vorzubereiten, mit deren Hilfe die komplette Arbeit nochmals vorgestellt und verteidigt wird. Nachdem die Endnote gebildet und an das Prüfungsamt übermittelt wurde, erhält der Studierende eine Diplomurkunde und häufig zusätzlich ein Gutachten über den gesamten Arbeitsverlauf, womit das Gesamtprojekt als abgeschlossen gilt. Im Vergleich zu dem vorher beschriebenen Projekt ist nun wesentlich, dass die instanzübergreifende Arbeit mehrerer Personen stärker in den Mittelpunkt rückt sowie eine Verwaltung mehrerer Dokumente an Bedeutung gewinnt. Zusätzliche Systemfunktionen sind denkbar wie das Überwachen der Emailkorrespondenz oder eine Versionskontrolle der Dokumente oder bisher entwickelter Programmkomponenten (was in einem Workflowmanagement-System auch durch Einbindung von Informationen externer Softwareapplikationen unterstützt werden kann).

2.4.5. Entwicklung einer neuen Webseite

Die Entwicklung einer Webapplikation unterscheidet sich von dem vorhergehenden Anwendungsszenario darin, dass bewusst mehrere Teammitglieder mit unterschiedlichen Zuständigkeitsbereichen an einem Projekt gemeinsam arbeiten und auf die Arbeitsergebnisse der anderen Mitglieder angewiesen sind. Bedingt durch die Nähe zum Software Engineering sind dafür spezialisierte Projektmanagementmethoden bereits gut erforscht, weswegen die Entwicklung häufig nach einem festen Vorgehensmodell (Wasserfallmodell, Spiralmodell, V-Modell und andere) erfolgt. So finden sich als typische Phasen in der Regel eine Planungs-, Design-, Produktions-, Test- und Freigabephase, wobei einzelne Phasen aufeinander zurückwirken können und der Abschluss einer Phase meist durch die Fertigstellung eines konkreten Produkts gekennzeichnet ist. Innerhalb der Phasen finden sich unterschiedliche Akteure mit verschiedenen Nutzerrollen wie Projektmanager, Consultants, Informationsarchitekten, Screendesigner, Datenbanker, Programmierer, Sicherheits- und Netzwerkexperten, Testnutzer als auch spätere Autoren und Anwender wieder. Während in der Planungsphase zunächst Gespräche mit den Kunden nötig sind (bspw. Experteninterviews), um nötige Anforderungen festzustellen und darauf aufbauend eine Zeit- und Ressourcenplanung durchführen zu können, folgt in der Designphase die konkrete Beschreibung des Funktionsumfangs und die Spezifikation eines Anwendungsmodells, wie die Applikation später zu implementieren ist. Ein erstes Screendesign wird erstellt, Navigationspfade werden identifiziert, eine Datenrepräsentation wird gewählt, Beziehungen zwischen einzelnen Modulen werden identifiziert und eine Performanceanalyse durchgeführt. Unterschiedliche Aufgaben werden dabei von verschiedenen Spezialisten übernommen – so ist ein Programmierer im Weiteren für den internen Programmfluss zuständig, während sich der Screendesigner um das Layout der Webseite kümmert.

Nachdem das in der Spezifikation festgelegte Modell in einem Prototyp umgesetzt wurde, werden fortwährend Tests durchgeführt, deren Ergebnisse auf den aktuellen Stand der Programmierung zurückwirken. Nach endgültiger Fertigstellung der Webseite erfolgt ein Roll-Out, wobei die Webapplikation in ihrer realen Umgebung veröffentlicht wird und dort praktisch im Einsatz ist. Dieses idealisierte Szenario umfasst in der Praxis eine Reihe weiterer Aspekte (Nutzerschulung, Wartungsverträge, Weiterentwicklung, Optimierung und andere) und lässt sich auf eine Reihe weiterer Ingenieursbereiche übertragen.

2.4.6. Workshop organisieren

In der Praxis werden Projektmanagementsysteme häufig dazu eingesetzt, langfristige Projekte mit mehreren Beteiligten besser koordinieren und deren Ausführungsstatus überwachen zu können. Dass dies nicht nur Softwareprojekte oder unternehmensinterne Aufgabenstellungen umfasst, soll das abschließende Anwendungsszenario zeigen. In diesem soll durch eine größere Personengruppe ein Wochenendseminar organisiert werden. Alternativ könnte dies ebenso gut ein Vortragsabend, eine Messe oder eine Fachtagung sein. In jedem Fall ist bedingt durch die Komplexität der Organisation eine Aufgabenteilung nötig, in der jede Person eigenständig für ihren Aufgabenteil zuständig ist, über den Organisationsfortschritt der Anderen aber jederzeit informiert sein muss, um doppelte Arbeiten oder Lücken zu vermeiden, und zusätzlich eine einfache Kommunikation und Absprache möglich sein sollte. Neben einem Hauptverantwortlichen („Projektleiter“) sind in diesem Fall Personengruppen identifizierbar, welche sich um die Programmorganisation und das Einladen der Referenten und Gäste kümmern, Verantwortliche für Catering und Übernachtung, Finanzkoordinatoren, Presseverantwortliche sowie eine Reihe externer Mitarbeiter.

2.5. Evaluierung gängiger Systeme

2.5.1. Herangehensweise

Dem Wortursprung zufolge (franz. „évaluer“ – bewerten) kennzeichnet eine Evaluierung im Allgemeinen „eine Beschreibung, Analyse und Bewertung von Prozessen und Produkten“ hinsichtlich vorher festgelegter Kriterien zur Sammlung von Grundlagendaten zur Entscheidungshilfe. Die nachfolgende Evaluierung soll dazu dienen, einen Überblick über den Funktionsumfang aktuell verfügbarer Systeme zu bekommen, die sich in der Praxis bewährt haben. Gleichzeitig soll eine Vergleichsgrundlage anhand auffälliger Nachteile und Schwächen dieser Systeme gegenüber eines neu entwickelten, semantikbasierten Systems geschaffen werden, welches Aspekte des Workflow- und Projektmanagements miteinander verbindet. Da sowohl Applikationen im Projektmanagement als auch im Workflowmanagement in einer Vielzahl vorhanden sind, wird die Evaluierung auf webbasierte Managementsysteme begrenzt, da davon auszugehen ist, dass Desktopanwendungen (wie Microsoft Projekt) einen abermals gesteigerten Funktionsumfang bieten, der mit der Implementierung einer prototypischen Webanwendung nicht erreicht werden kann. Vielmehr geht es in der Evaluierung darum, sich einen Überblick über vorhandene Systeme zu verschaffen als auch über darin verwendete Konzepte, Darstellungs- und Navigationsformen, die sich in der Praxis bewährt haben.

Um die Evaluierung durchzuführen, wurden zunächst alle verfügbaren Applikationen im Juli 2007 getrennt nach Projektmanagementsystemen und Workflowmanagementsystemen recherchiert. Um in die Testliste aufgenommen zu werden, musste die jeweilige Applikation öffentlich zugänglich sein oder zumindest eine Demoversion zu Testzwecken existieren. Als Quellen für verfügbare Managementapplikationen wurden gängige Softwarekataloge im Internet wie www.dmoz.org/Computers/Software/, www.softguide.de, www.pm-software.info oder www.workflowdownload.com genutzt. Aus allen gefundenen Produkten wurden mehrere potentiell interessante Vertreter ausgewählt, die als praxisrelevant eingestuft wurden und unter einem Kriterienkatalog aus Nutzersicht miteinander verglichen wurden. Die Evaluierungskriterien sind in Tabelle 2 näher aufgeschlüsselt.

Produktname	Offizielle Bezeichnung der Applikation
Generelle Informationen	
Hersteller	Herstellerangaben, sofern vorhanden
Version	Angaben zur getesteten Produktversion, damit bedingt Aussage zu Reife des Produktes möglich
Webseite	Verweis auf Entwicklerwebsite mit Downloadbereich oder Testumgebung
Entwicklungsbeginn	Sofern ermittelbar, Zeitraum in dem mit Entwicklung der Anwendung begonnen wurde (Indikator über Fortgeschrit-

	tenheit oder Neuartigkeit des Produkts)
Letzte Aktualisierung	Letzte Produktaktualisierung, Indikator inwieweit Anwendung noch weiterentwickelt wird oder zum Erliegen gekommen ist
Managementsystem-Kategorie	Projektmanagement, Workflowmanagement oder Groupware-Anwendung
Plattform	Auf welchen Systemplattformen kann die Anwendung genutzt werden, sofern nicht webbasiert
Programmiersprache	Implementierungssprache
Datenmodell	Sofern ein standardisiertes Datenaustauschformat vorhanden, wird dies hier benannt
Addons / Plugins vorhanden	Existieren Erweiterungsmöglichkeiten und zusätzliche Module für das Produkt
Sprachen	Angegebene Sprachpakete für das Nutzerinterface
Lizenz	Hinweise zur Lizenzierung des Produkts
System	
Funktionsgruppen	Grobbeschreibung der primären Systembereiche
Navigationsstruktur / Bedienkonzept	Beschreibung der zugrunde liegenden Navigation und der Art der Bedienung
Layout	Beschreibung des Seitenaufbaus und verwendeter Layoutkonzepte
Intuitive Bedienbarkeit	Wie einfach und intuitiv ist das vorliegende System (subjektiv) zu bedienen
Graphische Modellierwerkzeuge	Existieren Hilfswerkzeuge, um Projekte und Prozesse in einer Art Brainstorming spontan zu modellieren oder geschieht alles über ein textbasiertes Interface
Modulübergreifende Datenauswertung	Können auch Prozessdaten aus anderen Anwendungen importiert und verarbeitet werden
Nutzerverwaltung	Existiert eine Nutzer- und Rechteverwaltung?
Aufgabenverwaltung	Können neue Aufgaben erstellt und delegiert werden
Strukturierung von Aktivitäten	Basiert die Aufgabenverwaltung auf der Beschreibung einer Aufgabe (klassischer Projektansatz) oder können Aufgaben hierarchisch weiter strukturiert werden (Workflow-Konzept)
Dokumentenverwaltung	Können Projekten oder Aufgaben Dokumente zugeordnet werden
Zeit- und Forschrittsauswertungen	Existieren Hilfsmittel, welche den Fortschritt eines Projektes grafisch darstellen
Systembenachrichtigungsfunktionen	Wird de Nutzer über aktuelle und anstehende Aktivitäten vom System informiert
Kommunikationsfunktionen	Gibt es weitergehende Emailfunktionen und Kontaktmöglichkeiten mit anderen Nutzern
Weitere Groupware-Funktionen	Welche zusätzlichen Funktionen werden angeboten, die nicht unmittelbar dem Projekt- oder Workflowmanagement zuzuordnen sind
Anwendungsszenarien	
Szenario 1: Spaghetti kochen	Können einfache Prozesse in dem System abgebildet

	werden?
Szenario 2: Abendessen mit Freunden	Kann ein einfach strukturiertes Projekt in dem System abgebildet werden?
Szenario 3: Diplomarbeit schreiben	Können umfangreichere Projekte mit mehreren Beteiligten effizient mit der Anwendung verwaltet werden?
Szenario 4: Applikationsentwurf	Sind praktische Anwendungsszenarien mit höherer Komplexität und mehreren Beteiligten, Aufgaben und Dokumenten ohne Aufwand beherrschbar?
Szenario 5: Workshop organisieren	Können Großprojekte über einen längeren Zeitraum sinnvoll verarbeitet werden?
Gesamteindruck	
Hilfreiche Funktionen	Welche Funktionen sind positiv aufgefallen, welche in anderen Testkandidaten nicht zu finden sind
Fehlende Funktionen	Welche Funktionen fehlen, die gegen eine Verwendung des Testprodukts sprechen
Bewertung	Abschließende Bewertung des Gesamtsystems

Tabelle 2: Kriterienkatalog für die Evaluierung von Workflow- und Projektmanagementsystemen

2.5.2. Projektmanagementsysteme

2.5.2.1. Ausgewählte Testkandidaten

Zur Evaluierung aktueller Projektmanagementsysteme wurden die Produkte PHProjekt, Double Choco Latte, dotproject, WebCollab, PHPCollab und Basecamp ausgewählt.

2.5.2.2. PHProjekt

PHProjekt ist eine seit 2000 durch die Mayflower GmbH in München entwickelte Groupware Suite. Nach eigenen Angaben auf der Projektwebseite[7] zählt PHProjekt zu den besten freien Groupwaresystemen, die ein webbasiertes Projektmanagement unterstützen. Nicht zuletzt ist dies darin begründet, dass PHProjekt eine Vielzahl an Datenbankanbindungen unterstützt und ein ausgefeiltes Layouttemplate-System sowie eine feingranulare Rollenverwaltung bietet und darüber hinaus eine Reihe an Addons zur Verfügung stehen, welche die Integration anderer Applikationen problemlos ermöglichen sollen.

Der Fokus von PHProjekt liegt in erster Linie im Groupware-Bereich. Die Erstellung und Verwaltung verschiedener Projekte wird zwar vielfältig unterstützt, ist im Gesamtsystem aber dem Informationsaustausch zwischen verschiedenen Benutzern und weiteren Planungsfunktionen untergeordnet und letztendlich auf ein eigenständiges Modul beschränkt. Daneben finden sich vielfältige Funktionen zum Verwalten von Kontakten oder anderen Projektbeteiligten, Aufgabenlisten, Terminen, Dateien und Notizen, aber auch die Protokollierung bisheriger Arbeitsstunden wird ermöglicht. Zusätzlich stehen groupware-typische Module wie ein Forum oder Chat zur Verfügung.

[7] Siehe http://www.phprojekt.de/index.php?name=presse

Projekte lassen sich hierarchisch in Subprojekte untergliedern. Ebenso können Dateien bestimmten Projekten zugeordnet werden und in einer Ordnerstruktur hierarchisch verwaltet werden. Analog können einem Projekt beteiligte Personen und Verantwortliche zugeordnet werden. Sämtliche Zuordnungen werden auf einer Übersichtsseite tabellarisch dargestellt, in die der aktuelle Benutzer involviert ist. Durch Filter auf einzelne Tabellenspalten kann die Anzeige eingeschränkt werden. Das gesamte System ist dahingehend einfach benutzbar, wirkt allerdings im Standard-Layout sehr textlastig und scheint trotz der Filterfunktionen bei umfangreicheren Projekten schnell an Übersichtlichkeit zu verlieren, sofern die Strukturierungsmöglichkeiten des Systems nicht zu jeder Zeit ausgenutzt werden. Eine spontane Modellierung gestaltet sich schwierig, da durch die Eingabemasken des Systems ein enger Rahmen vorgegeben wird, andererseits nur wenige Automatismen existieren oder sogar eine grafische Übersicht, welche Eingaben verschiedener Module zueinander in Beziehung setzen würde. Dazu kommt, dass Projekte mit PHProjekt zwar verwaltet werden können, darüber hinausgehend eine Verwaltung oder Strukturierung von Aktivitäten in diesen Projekten nur rudimentär vorhanden und möglich ist. So können zwar einem Projekt einzelne Aufgaben zugeordnet werden, welche jedoch nur eine textuelle Beschreibung darstellen und durch einen Endtermin terminierbar sind. Damit sind Engineering-Aufgaben wie ein Bug-Tracking zwar einfach realisierbar, die unter 2.4. genannten Anwendungsszenarien aber nur eingeschränkt abbildbar.

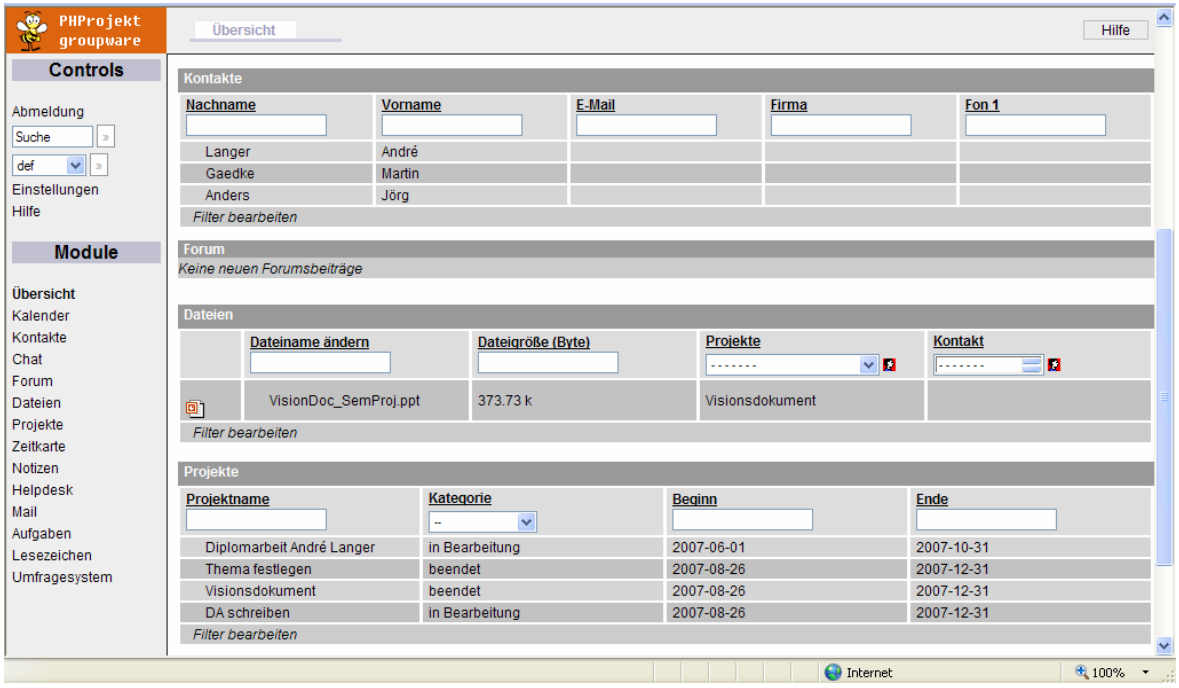


Abbildung 6: Screenshot PHProject

2.5.2.3. Double Choco Latte

Unter dem Namen Double Choco Latte findet sich im World Wide Web ein weiteres webbasiertes Projektmanagement-System, welches unter der GNU General Public License steht und neben der Projekt- und Auftragsverwaltung zahlreiche Funktionen zur Zeit- und Statuserfassung bzw. –auswertung bietet. Im Gegensatz zu PHProjekt handelt es sich dabei weniger um ein Groupware-System als vielmehr um ein System zur Verwaltung von Arbeitsaufträgen (Work Orders) im Rahmen verschiedener Projekte. Dementsprechend steht bei Double Choco Latte vor allem die Bearbeitung empfangener Anfragen im Mittelpunkt, die sowohl nach Priorität und aktuellem Status, als auch nach Schweregrad verwaltet werden können. Anfragen von Kunden, die in Organisationen zusammengefasst werden können, werden über ein integriertes Ticketsystem gemanaged. Als Folge dessen wird auch bei diesem Produkt bei der Umsetzung der Beispielszenarien schnell ersichtlich, dass Double Choco Latte eher als ein Bugtrackingtool einer bestehenden Softwareanwendung konzipiert ist, als dass es ein universelles Projektmanagementsystem darstellen würde, mit welchem auch abstraktere und IT-fremde Projekte verwaltet werden könnten.

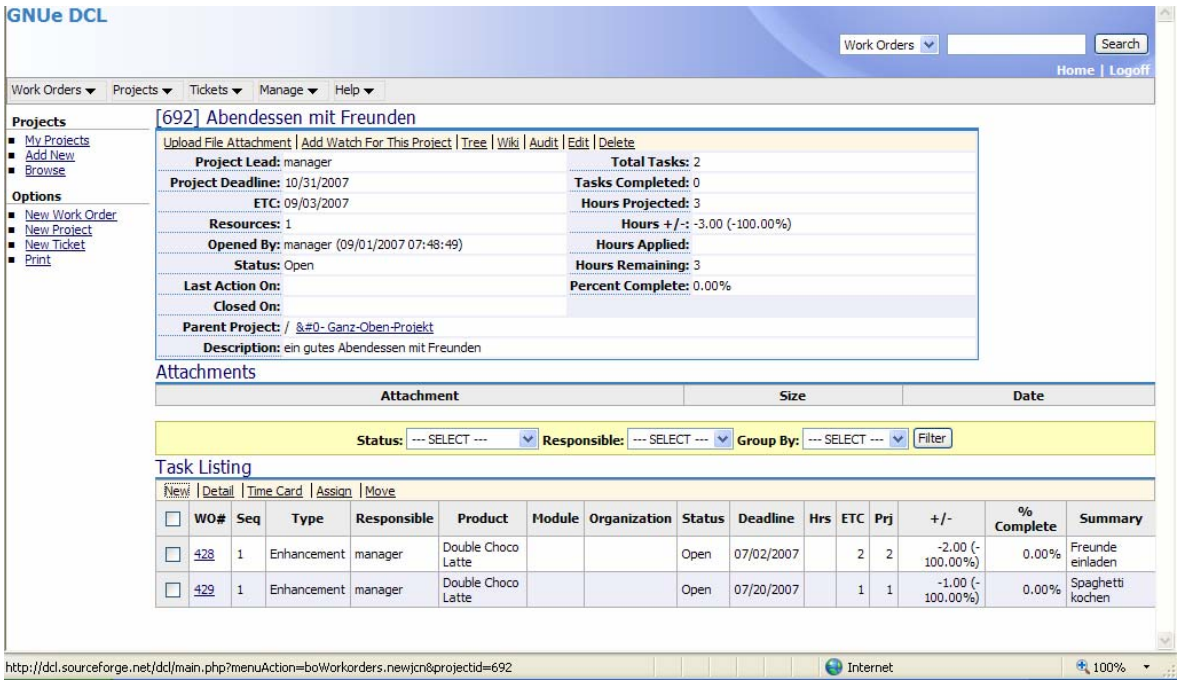


Abbildung 7: Screenshot Double Choco Latte

2.5.2.4. dotproject

Nach einer Beschreibung auf der Entwicklerwebseite[8] ist DotProject ein seit 2000 entwickeltes „state-of-the-art“ Projektmanagementsystem, welches als Alternative zu Microsoft Project oder anderen kommerziellen Softwareprodukten entwickelt wurde und (im Gegensatz zu anderen kollaborativen Systemen und Groupwarelösungen) eine vollwertige Projektmanagement bieten soll. Besonders interessant ist dabei der Ansatz, seit Beginn nicht eine von vielen Groupwarelösungen zu entwickeln, sondern eine universelle und einfach zu bedienende Projektmanagementumgebung anzustreben. Dieses Konzept merkt man in der praktischen Arbeit mit dotProject schnell, denn es findet sich eine Reihe essentieller Funktionen, welche in anderen webbasierten Projektmanagementsystemen fehlen. So können Projekte und darin enthaltene Aufgaben nicht nur hierarchisch untergliedert werden, sondern auch spezielle Ereignisse (events) separat eingetragen werden. Diese Ereignisse erinnern bereits an ein Konzept aus dem Workflow-Bereich, beinhalten letztendlich aber nur eine explizite Kennzeichnung von projektbezogenen Terminen oder Treffen, welche in einem Kalendermodul eingetragen werden. Ebenso wird in der Eingabemaske zur Erstellung einer neuen Aufgabe nicht nach einem Bearbeitungszeitraum gefragt, sondern zunächst nur um den Inhalt der Aufgabe und ob es sich um einen Meilenstein im Rahmen des Projekts handelt. Auch dies lässt erkennen, dass dotproject von Grund auf ein schlüssiges Projektmanagementvokabular verwendet. So lassen sich Aufgaben in einem weiteren Modul problemlos umsortieren oder zwischen Projekten verschieben. Alle eingetragenen Aufgaben können anschließend in Gantt-Diagrammen dargestellt oder in einem Report ausgewertet werden. Dotproject ist damit zusammenfassend ein sehr professionelles Projektmanagementsystem mit vielen Funktionen, auch wenn an einigen Stellen im System die Spezialisierung auf den Geschäftsbereich (Unterteilung in administrative und operative Aufgaben) erkennbar ist.

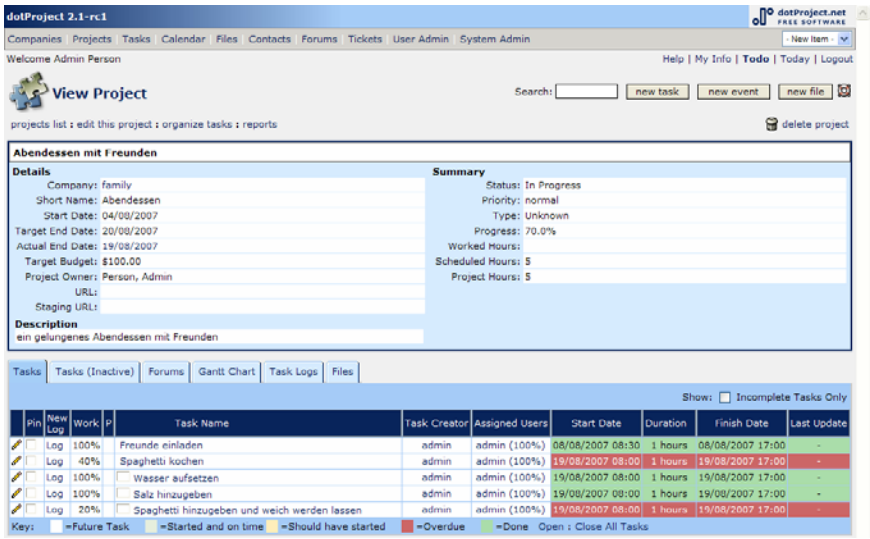

dotProject 2.1-rc1
Companies Projects Tasks Calendar Files Contacts Forums Tickets User Admin System Admin
Welcome Admin Person
Help | My Info | Todo | Today | Logout
View Project
Search:
new task
new event
new file
projects list : edit this project : organize tasks : reports
delete project
Abendessen mit Freunden
Details
Company: family
Short Name: Abendessen
Start Date: 04/08/2007
Target End Date: 20/08/2007
Actual End Date: 19/08/2007
Target Budget: $100.00
Project Owner: Person, Admin
URL:
Staging URL:
Summary
Status: In Progress
Priority: normal
Type: Unknown
Progress: 70.0%
Worked Hours:
Scheduled Hours: 5
Project Hours: 5
Description
ein gelungenes Abendessen mit Freunden
Tasks Tasks (Inactive) Forums Gantt Chart Task Logs Files
Show: Incomplete Tasks Only
Pin New Log Work P Task Name Task Creator Assigned Users Start Date Duration Finish Date Last Update
Log 100% Freunde einladen admin admin (100%) 08/08/2007 08:30 1 hours 08/08/2007 17:00
Log 40% Spaghetti kochen admin admin (100%) 19/08/2007 08:00 1 hours 19/08/2007 17:00
Log 100% Wasser aufsetzen admin admin (100%) 19/08/2007 08:00 1 hours 19/08/2007 17:00
Log 100% Salz hinzugeben admin admin (100%) 19/08/2007 08:00 1 hours 19/08/2007 17:00
Log 20% Spaghetti hinzugeben und weich werden lassen admin admin (100%) 19/08/2007 08:00 1 hours 19/08/2007 17:00
Key: =Future Task =Started and on time =Should have started =Overdue =Done Open : Close All Tasks

Abbildung 8: Screenshot dotProject

[8] Vgl. http://www.dotproject.net/modules.php?op=modload&name=News&file=article&sid=5

2.5.2.5. WebCollab

Ein weiteres unter der GPL lizensiertes Projektmanagementwerkzeug ist WebCollab, welches seit 2002 unter sourceforge.net stetig weiterentwickelt wird. Die Produktbeschreibung, wonach WebCollab einfach zu benutzen sei und Nutzer dazu animiere, gemeinsam an einem Projekt zu arbeiten ohne kompliziert oder grafisch aufwändig zu sein, klingt zunächst wenig spektakulär. Umso interessanter stellt sich WebCollab im praktischen Einsatz heraus, wo einige Ideen im zugrunde liegenden Systemkonzept positiv auffallen und die Modellierung und Bedienung erleichtern. So steht nicht nur das Projektmanagement an sich im Mittelpunkt des Systems, sondern es können auch den Projekten zugeordnete Aufgaben beliebig strukturiert werden. Änderungen an Projektdetails oder am Projektstatus werden hervorgehoben und sind so für alle Nutzer ersichtlich. Nutzer wiederum werden in (beliebigen) Nutzergruppen zusammengefasst, wodurch einzelne Personen gemeinsam verwaltet werden können. Unabhängig davon existiert ein Rechtesystem, um Nutzer administrative Rechte zuzuteilen oder zu entziehen. Dennoch bleibt die Bedienung von WebCollab einfach und die Bildschirmdarstellung übersichtlich strukturiert.

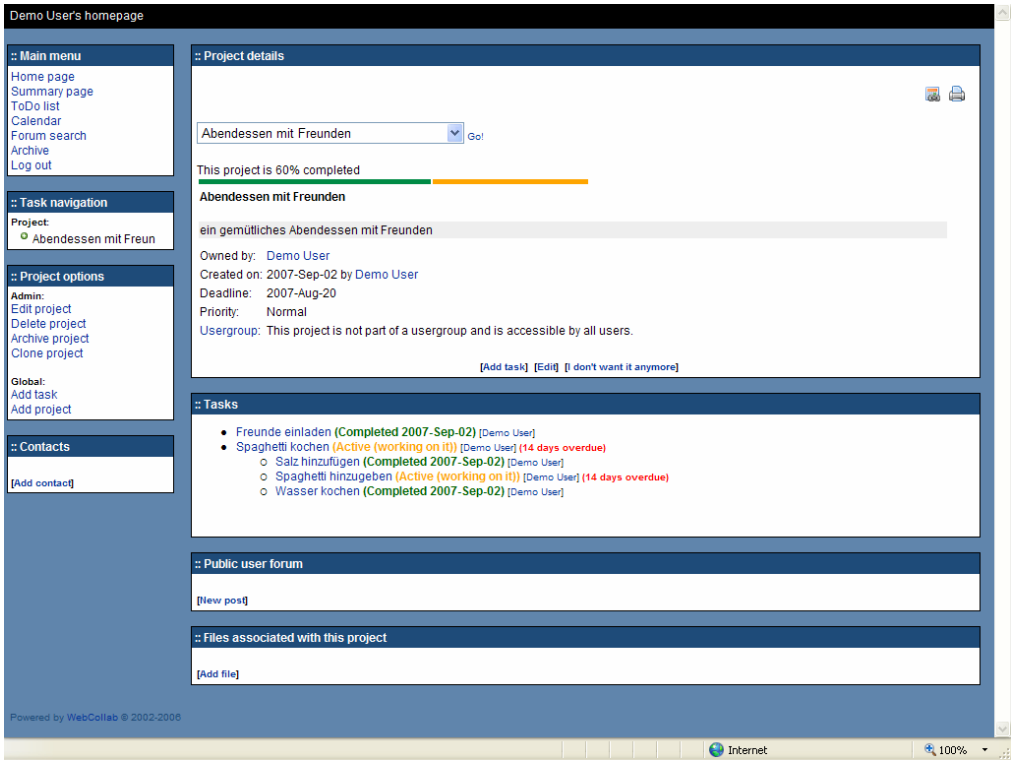


Abbildung 9: Screenshot WebCollab

2.5.2.6. PHPCollab

PHPCollab ist ein kollaboratives System, dessen Entwicklung zwischenzeitlich eingestellt zu sein schien, momentan aber wieder weiterentwickelt wird, und für die Praxis eine Reihe interessanter Funktionalitäten bereitstellt. Exemplarisch seien getrennte Entwickler- und Kundenbereiche, ein Newsdesk, Berichtegeneratoren und ein Dokumentenmanagement genannt. Trotz das PHPColab kostenfrei angeboten wird, finden sich viele Unternehmen, welches dieses Produkt im professionellen Arbeitsalltag zur Projekt- und Auftragsabwicklung einsetzen.

Betont wird sowohl die kollaborative Arbeit in Team, bei der verschiedene Teammitglieder unterschiedliche Tätigkeitsschwerpunkte übernehmen, als auch die Trennung von Teamaufgaben und Kundenanfragen, indem öffentliche Projektseiten bereitgestellt werden können. Das Projektmanagement stellt die zentrale Komponente des Managementsystems dar, von dem aus auf weitere Module wie Aufgaben, Diskussionen, verknüpfte Inhalte und Teammitglieder sowie Notizen zugegriffen werden kann.

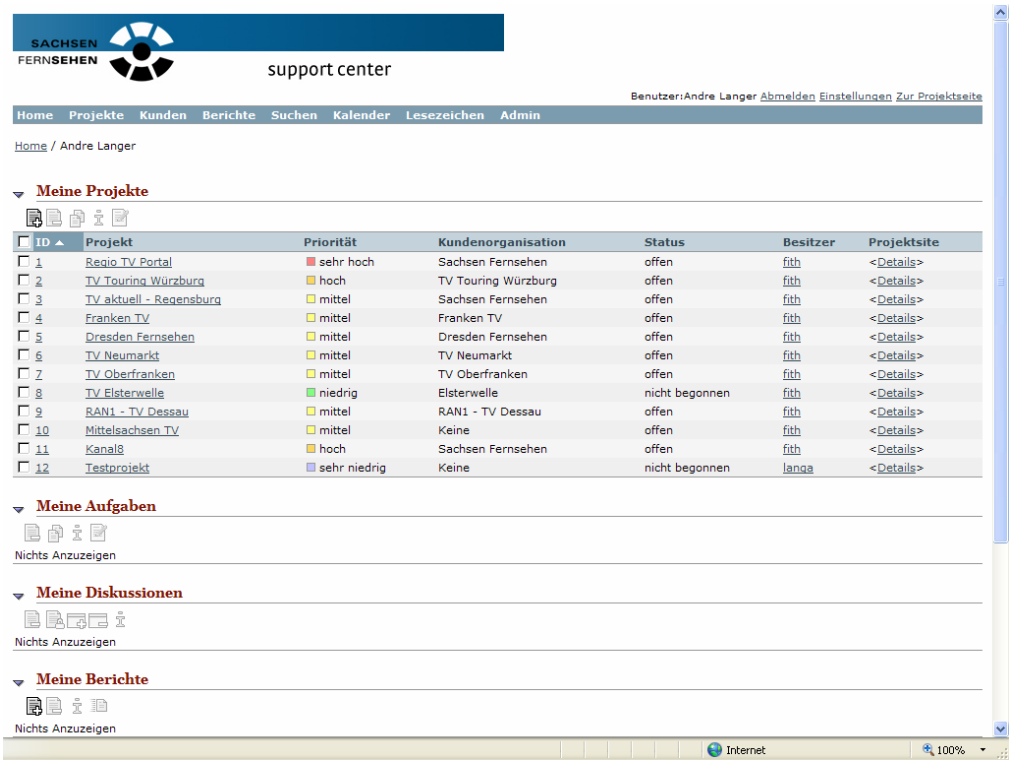

SACHSEN FERNSEHEN
support center
Benutzer:Andre Langer Abmelden Einstellungen Zur Projektseite
Home Projekte Kunden Berichte Suchen Kalender Lesezeichen Admin
Home / Andre Langer
Meine Projekte
ID Projekt Priorität Kundenorganisation Status Besitzer Projektsite
1 Regio TV Portal sehr hoch Sachsen Fernsehen offen fith <Details>
2 TV Touring Würzburg hoch TV Touring Würzburg offen fith <Details>
3 TV aktuell - Regensburg mittel Sachsen Fernsehen offen fith <Details>
4 Franken TV mittel Franken TV offen fith <Details>
5 Dresden Fernsehen mittel Dresden Fernsehen offen fith <Details>
6 TV Neumarkt mittel TV Neumarkt offen fith <Details>
7 TV Oberfranken mittel TV Oberfranken offen fith <Details>
8 TV Elsterwelle niedrig Elsterwelle nicht begonnen fith <Details>
9 RAN1 - TV Dessau mittel RAN1 - TV Dessau offen fith <Details>
10 Mittelsachsen TV mittel Keine offen fith <Details>
11 Kanal8 hoch Sachsen Fernsehen offen fith <Details>
12 Testprojekt sehr niedrig Keine nicht begonnen langa <Details>
Meine Aufgaben
Nichts Anzuzeigen
Meine Diskussionen
Nichts Anzuzeigen
Meine Berichte
Nichts Anzuzeigen
Internet
100%

Abbildung 10: Screenshot PHPCollab

2.5.2.7. BaseCamp

BaseCamp unterscheidet sich von den bisher vorgestellten Webapplikationen in zwei Punkten: Zum einen handelt es sich bei BaseCamp um ein System, welches kollaboratives Arbeiten im Kontext des Web 2.0 unterstützt und aus diesem Grund großen Gebrauch von AJAX design patterns macht, um eine möglichst einfache Bedienung zu ermöglichen. Zum anderen ist BaseCamp eine Webapplikation, welche seit 1999 von der Firma 37signals entwickelt wird und auch auf den Servern dieser Firma bereitgestellt wird – mit unterschiedlichen Zugangstarifen, von denen nur einer kostenfrei ist. BaseCamp steht unter keiner Open-Source-Lizenz und es existiert auch keine Version zur lokalen Installation. Inzwischen gibt es aber mehrere Anbieter, welche zu BaseCamp ähnliche Webapplikationen entworfen und ins Netz gestellt haben.[9] Wegen des ungewöhnlichen Projektmanagementansatzes und der Web 2.0 – Orientierung wurde BaseCamp in die Evaluierung aufgenommen und soll nachfolgend kurz vorgestellt werden.

Nachdem ein Zugang zu Basecamp eingerichtet wurde, können zunächst wie gewohnt neue Projekte über eine entsprechende Schaltfläche erstellt werden. Im Gegensatz zu andern Projektmanagementanwendungen ist dazu aber nur die Angabe eines Projektnamens nötig. Etwaige Start- und Endtermine müssen nicht angegeben werden. Projektbeteiligte können über eine Nutzerverwaltung zugeordnet und zu Basecamp zugeordnet werden. Die Projektoberfläche erinnert an eine Groupwarelösung, wobei typische Schaltflächen wie „neue Aufgabe hinzufügen“ fehlen. Stattdessen findet sich im Navigationsbereich ein To-Do-Modul, ein Meilensteinmodul sowie ein Whiteboardmodul. Nach dem „Keep it Simple and Stupid“ (KISS)-Prinzip) können darüber ToDo-Listen angelegt werden, in welche projektrelevante Aktivitäten und ein Hauptverantwortlicher eingetragen werden können. Diese sind ebenfalls nicht an ein Ausführungsdatum gebunden sondern bieten lediglich eine Statusmarkierung, ob sie bereits durchgeführt worden. Für datumsrelevante Ereignisse steht unter BaseCamp das Meilensteinkonzept zur Verfügung, wodurch terminierte Ereignisse gekennzeichnet und eingetragen werden können. Schließlich existiert mit dem Whiteboard eine Art Wiki, worüber mehrere Personen an einem gemeinsamen Datenbestand arbeiten und verschiedene Dokumentversionen miteinander vergleichen können. Im Vergleich zu klassischen Projektmanagementsystemen ist dies eine ungewöhnliche Systemarchitektur, da statt einer Managementsoftware mit vielen Abhängigkeiten ein Produkt angeboten wird, welches sehr viele Freiheiten erlaubt. Die Nutzungsstatistiken von BaseCamp mit bisher über 700.000 erstellten Projekten[10] scheint aber für diese Idee zu sprechen.

[9] Als Beispiel sei ActiveCollab genannt (www.activecollab.com)

[10] Quelle: http://www.basecamphq.com/examples, Stand: 27.08.2007 18:26 Uhr

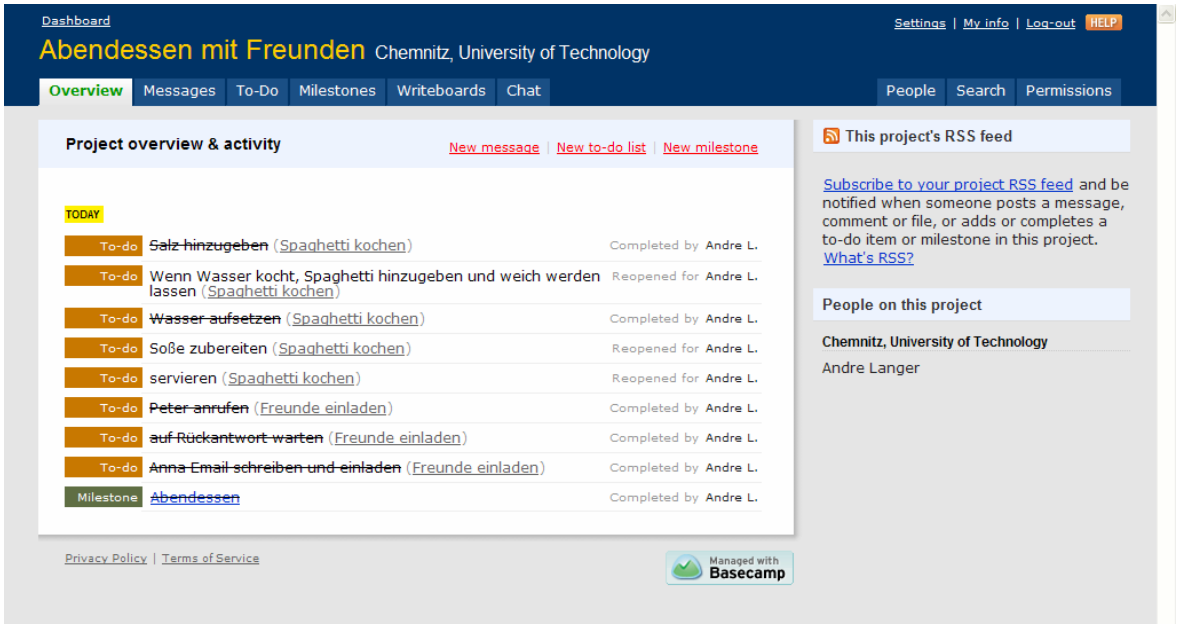

Dashboard
Settings | My info | Log-out HELP
Abendessen mit Freunden Chemnitz, University of Technology
Overview Messages To-Do Milestones Writeboards Chat People Search Permissions
Project overview & activity
New message | New to-do list | New milestone
TODAY
To-do ~~Salz hinzugeben~~ (Spaghetti kochen) Completed by Andre L.
To-do Wenn Wasser kocht, Spaghetti hinzugeben und weich werden lassen (Spaghetti kochen) Reopened for Andre L.
To-do ~~Wasser aufsetzen~~ (Spaghetti kochen) Completed by Andre L.
To-do Soße zubereiten (Spaghetti kochen) Reopened for Andre L.
To-do servieren (Spaghetti kochen) Reopened for Andre L.
To-do ~~Peter anrufen~~ (Freunde einladen) Completed by Andre L.
To-do ~~auf Rückantwort warten~~ (Freunde einladen) Completed by Andre L.
To-do ~~Anna Email schreiben und einladen~~ (Freunde einladen) Completed by Andre L.
Milestone ~~Abendessen~~ Completed by Andre L.
Privacy Policy | Terms of Service
Managed with Basecamp
This project's RSS feed
Subscribe to your project RSS feed and be notified when someone posts a message, comment or file, or adds or completes a to-do item or milestone in this project.
What's RSS?
People on this project
Chemnitz, University of Technology
Andre Langer

Abbildung 11: Screenshot BaseCamp

2.5.3. Workflowmanagementsysteme

2.5.3.1. Ausgewählte Testkandidaten

Bei der Auswahl geeigneter Workflowmanagementsysteme fiel schnell auf, dass Anwendungen, mit welchen Workflows einfach abgebildet und dokumentiert werden können, im Internet schwieriger zu finden sind als die Fülle an Projektmanagementsystemen. Die geringe Auswahl an frei verfügbaren Workflowmanagementapplikationen war darüber hinaus fast durchgängig in Java implementiert und auf XPDL basierend. Daraus resultierend zerfielen viele der angebotenen Systeme in eine reine Workflow Engine, welche eine Prozessdefinition als Eingabedatei annimmt und den darin definierten Prozess ausführt, und in einen grafischen Workflow Editor, der zur Erstellung der Workflow-Beschreibung diente. Eine webbasierte Umsetzung davon war aber eher die Ausnahme. Letztendlich wurden daher der JaWE Java XPDL Editor, Bonita, Imixs, OSWorkflow, Runa und Oryx getestet.

2.5.3.2. JaWE Java XPDL Editor

Als erstes Produkt wird der Java XPDL Editor vorgestellt, der in der Enydra Community unter der LGPL frei angeboten und weiterentwickelt wird. An sich handelt es sich dabei um einen reinen Workflow-Editor, welcher Workflowbeschreibungen als XPDL erzeugt, welche aber an eine Workflowengine, wie beispielsweise Enhydra Shark aus dem gleichen Projekt, übergeben werden können. Der Editor ist Java-basiert und bietet eine intuitiv bedienbare Oberfläche zur Erstellung von Packages und zur Modellierung von Prozessen.

Prozesse bestehen dabei aus Start- und Endpunkten, Aktivitäten und Übergängen zwischen zwei Aktivitäten. Aktivitäten können für die Workflow-Engine abstrakt oder toolbasiert sein. Es werden vielfältige Möglichkeiten zur Definition von Prozessvariablen und zur Erstellung von Aktivitäten geboten, welche eine Reihe von Variablen eines bestimmten Typs als Eingabe nehmen und das Interaktionsergebnis an die nächste Aktivität weitergeben können. Außerdem besteht die Möglichkeit, Aktivitäten zu so genannten activity sets zu gruppieren oder bereits definierte Prozesse als Subprozesse hierarchisch einzubinden. Abschließend bietet der Editor die Möglichkeit, die erstellten XPDL – Beschreibungen auf Validität zu prüfen. Inzwischen existieren darüber hinaus weitere Entwicklungsgruppen, die auf dem JaWE aufbauend erweiterte Funktionen bieten.[11]

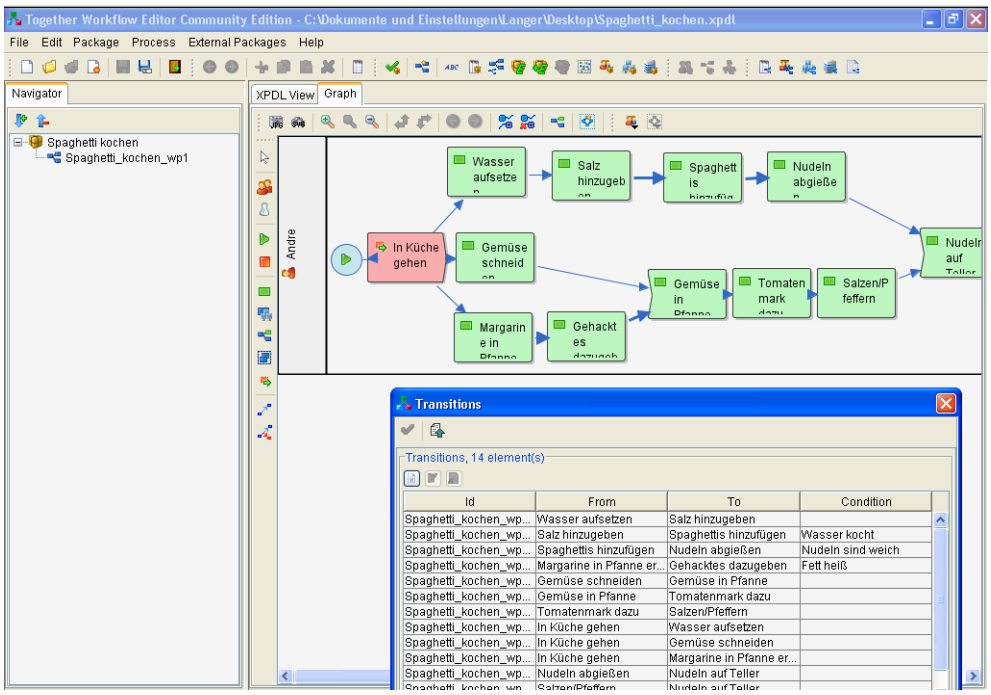


Abbildung 12: Screenshot JaWE Java XPDL Editor

[11] Siehe bspw. JPEd (http://jped.sourceforge.net/cms/index.php)

2.5.3.3. Bonita

„Workflow made graphical“ – mit diesem Slogan wirbt das Projekt Bonita auf seiner Internetseite.[12] Insgesamt umfasst die Workflow-Lösung drei wesentliche Komponenten: eine Workflow Engine zur Ausführung von XPDL-Beschreibungen, einen Workflow-Editor als auch einen XForms-Editor. Implementiert ist das Workflowsystem in Java und läuft auf dem Jonas Application Server.

Bei einem ersten Testlauf des Systems fällt als erstes die Integration aller Systemkomponenten unter einer gemeinsamen Oberfläche auf. Nach einer Authentifizierung über ein Anmeldeformular wird eine übersichtliche Navigation bereitgestellt, welche das Design eines Workflows, die Ausführung eines Workflows als auch die Überwachung bisheriger Workflow- und Systemaktivitäten ermöglicht. Daneben existieren Administrationsfunktionen zur Nutzer- und Systemverwaltung. Der Workflow- und XForms-Editor erscheint dabei als eigenständig entwickeltes Projekt (ProEd), mit welchem einfache Workflow-Beschreibungen in XPDL erstellbar sind. Diese können anschließend in Bonita gestartet und durch verschiedene, dem System bekannte, Nutzer ausgeführt werden. Eine kollaborative Arbeit an einem gemeinsamen Workflow wird dabei ohne zusätzlichen Aufwand unterstützt. Weiterhin existieren Web Service APIs, welche die Nutzung auch über Systemgrenzen hinweg ermöglichen. Die Ausführung eines Workflows an sich ist zwar weiter formularbasiert und der bisherige Prozessfortschritt nicht grafisch dargestellt, allein die Modellierung des Workflows innerhalb des Gesamtsystems mithilfe eines grafischen Editors unterstreicht aber den Slogan „Workflow made graphical“.

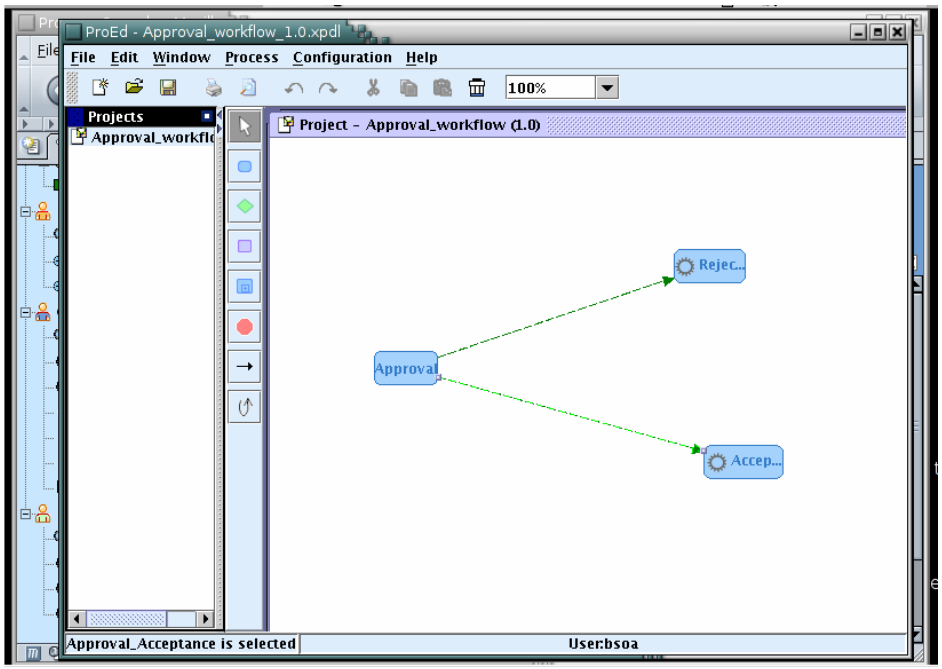


Abbildung 13: Screenshot Bonita

[12] Vgl. http://wiki.bonita.objectweb.org/xwiki/bin/view/Main/Downloads

2.5.3.4. Imixs IX JEE Workflow

Die Imixs Software Solution GmbH mit Hauptsitz in München bietet seit Ende 2005 für den GlassFish Application Server ein Workflow-System an, welches unter der LGPL veröffentlicht wurde. Workflows werden dabei in einem eigenen imx-Dateiformat verwaltet, welches ein XML-Dokument mit Definition der enthaltenen Workflowelemente enthält. Dazu existiert ein Plugin (IX Workflow Modeler) für die Eclipse – Entwicklungsumgebung zur Modellierung von Workflows und zur Abspeicherung im imx-Format. Diese XML-Beschreibungen können anschließend durch die Workflow Engine verarbeitet werden, aber auch einfach nach XPDL konvertiert werden. Auf der Webseite steht unter Anderem eine Testumgebung bereit, mit welcher einfache Prozessbeschreibungen unter Zuhilfenahme von AJAX basierten Funktionen ausgeführt werden können. Daneben stellt das Workflowsystem von Imixs eine umfassende API und eine Menge an Web Services bereit, um das System beispielsweise in SOA (service-orientated architecture)-Umgebungen zu integrieren.

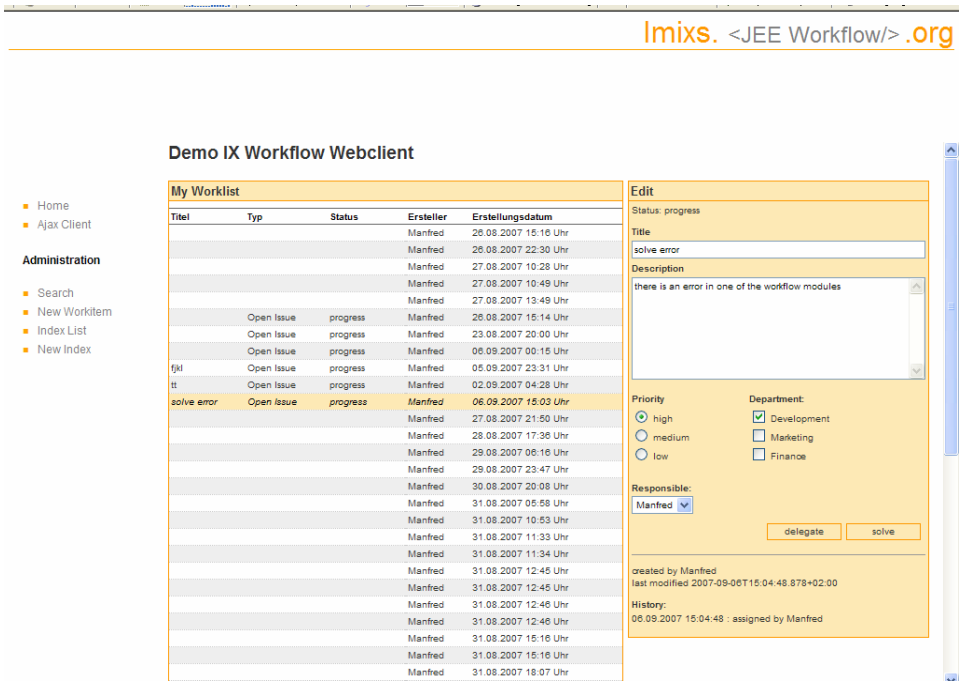


Abbildung 14: Screenshot Imixs

2.5.3.5. OSWorkflow

„*OSWorkflow is fairly different from most other workflow systems available […] (because) it is extremely flexible. […] For example, it does not mandate a graphical tool for developing workflows, and the recommended approach is to write the xml workflow descriptors 'by hand'*."[13] Mit dieser ungewöhnlichen Beschreibung beginnt die Produktbeschreibung von OSWorkflow, einem Java-basierten Workflowmanagementsystem, welches als eine „*low level workflow implementation*" angesehen werden soll. In der Tat umfasst das System eher ein Workflow-spezifisches, xml-basiertes Dokumentenformat sowie eine WorkflowEngine-API als eine vollwertige Webapplikation. Workflowbeschreibungen werden anschließend in Java instantiiert und auf einem Apache Tomcat zur Ausführung gebracht. Selbst die XML-Beschreibung eines Workflows ist an sich sehr einfach gehalten und umfasst im Wesentlichen nur die Konzepte Zustand und Aktion, wobei eine Aktion nicht eine Aktivität kennzeichnet, sondern einen Übergang zwischen zwei Zuständen in Verbindung mit bestimmten Bedingungen oder Funktionen. Ein ausgezeichneter Startzustand findet sich für jeden Workflow zur Instantiierung, ein Endzustand hingegen wird per se nicht benötigt. OSWorkflow ist damit insbesondere für erfahrene Java-Programmierer interessant, welche Workflows algorithmisch entwerfen möchten. Für einen schnellen Einsatz ohne Programmierkenntnisse ist OSWorkflow nicht geeignet (kennzeichnet in diesem Sinne eher eine Art framework), was das Entwicklerteam auch betont, darin aber den wesentlichen Vorteil einer maximalen Flexibilität sieht.

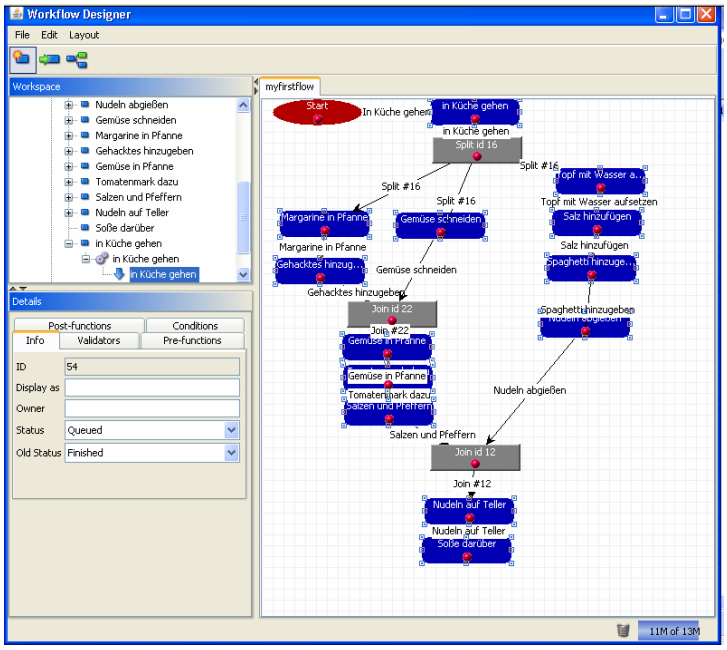


Abbildung 15: OSWorkflow

[13] Quelle: http://www.opensymphony.com/osworkflow/, Datum: 19.08.2007 14:43 Uhr

2.5.3.6. RunaWFE

Die Runa Workflow Engine ist eine Workflowengine, welche auf einer anderen Business Process Management (BPM)-Engine namens jBPM von JBoss (Red Hat) aufsetzt. Als Datenaustauschformat zur Repräsentation von Workflows wird nicht XPDL oder BPEL, sondern ein eigenes Format unter dem Namen jPdl[14] verwendet. Ähnlich wie bei anderen Workflow Engines ist der Zugriff auf die RunaWFE – Applikation nutzerabhängig und durch einen Anmeldedialog geschützt, über dessen Funktionen die Rollen des Nutzers im Workflow ermittelt werden. Auf einer Übersichtsseite werden anschließend alle zu bearbeitenden Aufgaben aufgelistet, in deren Workflows der angemeldete Nutzer involviert ist. Ebenso können nutzerabhängig neue Prozesse gestartet oder der Verlauf bisheriger Prozesse analysiert werden. Interessant ist vor allem der im jPdl-Format berücksichtige Versionierungsaspekt von Prozessen.

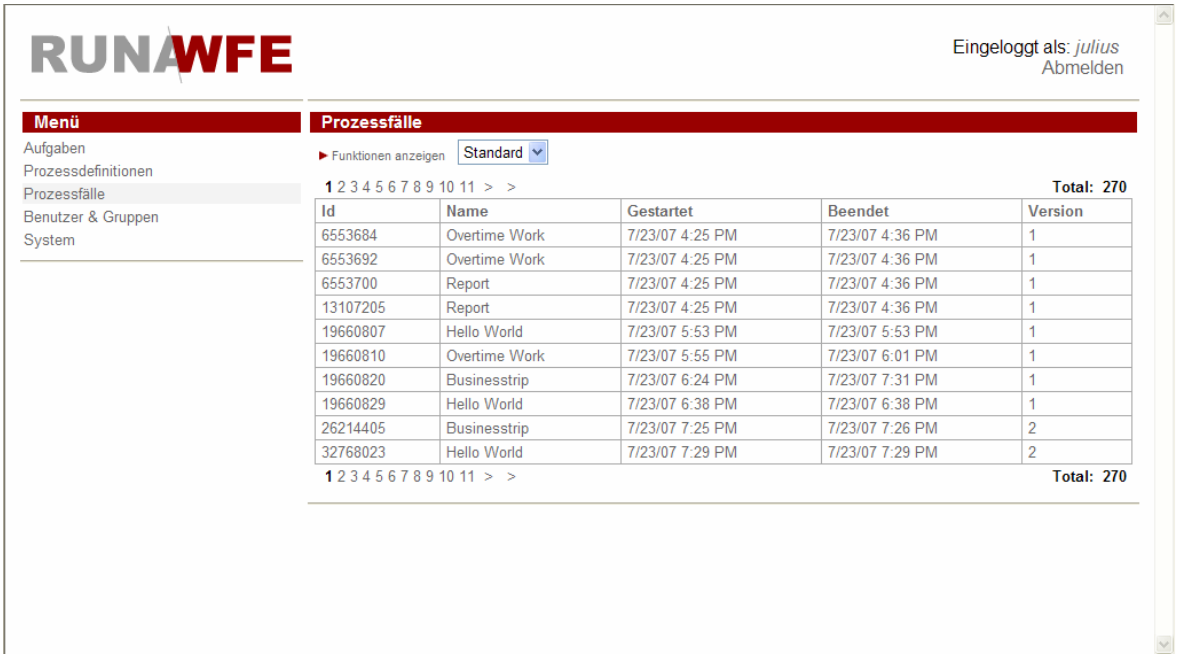

RUNAWFE
Eingeloggt als: julius
Abmelden
Menü
Aufgaben
Prozessdefinitionen
Prozessfälle
Benutzer & Gruppen
System
Prozessfälle
Funktionen anzeigen
Standard
1 2 3 4 5 6 7 8 9 10 11 > >
Total: 270
Id | Name | Gestartet | Beendet | Version
6553684 | Overtime Work | 7/23/07 4:25 PM | 7/23/07 4:36 PM | 1
6553692 | Overtime Work | 7/23/07 4:25 PM | 7/23/07 4:36 PM | 1
6553700 | Report | 7/23/07 4:25 PM | 7/23/07 4:36 PM | 1
13107205 | Report | 7/23/07 4:25 PM | 7/23/07 4:36 PM | 1
19660807 | Hello World | 7/23/07 5:53 PM | 7/23/07 5:53 PM | 1
19660810 | Overtime Work | 7/23/07 5:55 PM | 7/23/07 6:01 PM | 1
19660820 | Businesstrip | 7/23/07 6:24 PM | 7/23/07 7:31 PM | 1
19660829 | Hello World | 7/23/07 6:38 PM | 7/23/07 6:38 PM | 1
26214405 | Businesstrip | 7/23/07 7:25 PM | 7/23/07 7:26 PM | 2
32768023 | Hello World | 7/23/07 7:29 PM | 7/23/07 7:29 PM | 2
1 2 3 4 5 6 7 8 9 10 11 > >
Total: 270

Abbildung 16: Screenshot RunaWFE

[14] Spezifikation siehe unter http://www.jboss.com/products/jbpm/docs/jPdl

2.5.3.7. Oryx

Während sich einige webbasierte, kostenfreie Workflowmanagemensysteme finden, welche auf XPDL, jPdl oder ein anderes proprietäres XML-Austauschformat (bspw. die Simple Workflow Definition Language[15]) setzen, finden sich bisher überraschend wenige Webapplikationen, welche eine Modellierung in der Business Process Modeling Notation (BPMN) ermöglichen, die darauf aufbauend in die Business Process Execution Language (BPEL, oder ein anderes Format wie XPDL) übersetzt und von einer Workflow Engine ausgeführt werden kann.[16] Ein vielversprechendes Projekt in dieser Richtung wird seit 2006 an der Universität Potsdam entwickelt. Unter dem Namen Oryx ist dort ein webbasierter BPMN-Editor entstanden, der großen Gebrauch von Funktionen aus AJAX frameworks macht. Das Besondere an dem Editor ist die automatische Speicherung des erstellten Modells in dem HTML-Code der momentan aufgerufenen Webseite als Embedded RDF (eRDF). Durch diesen Ansatz ist es theoretisch möglich, das andere Applikationen den Workflowgraph problemlos aus dem HTML-Dokument extrahieren und ausführen können. Die offizielle Projektseite ist erst im Juli 2007 unter http://www.oryx-editor.org online gegangen, weswegen derartige Weiterentwicklungen in Zukunft sicherlich möglich sein könnten.

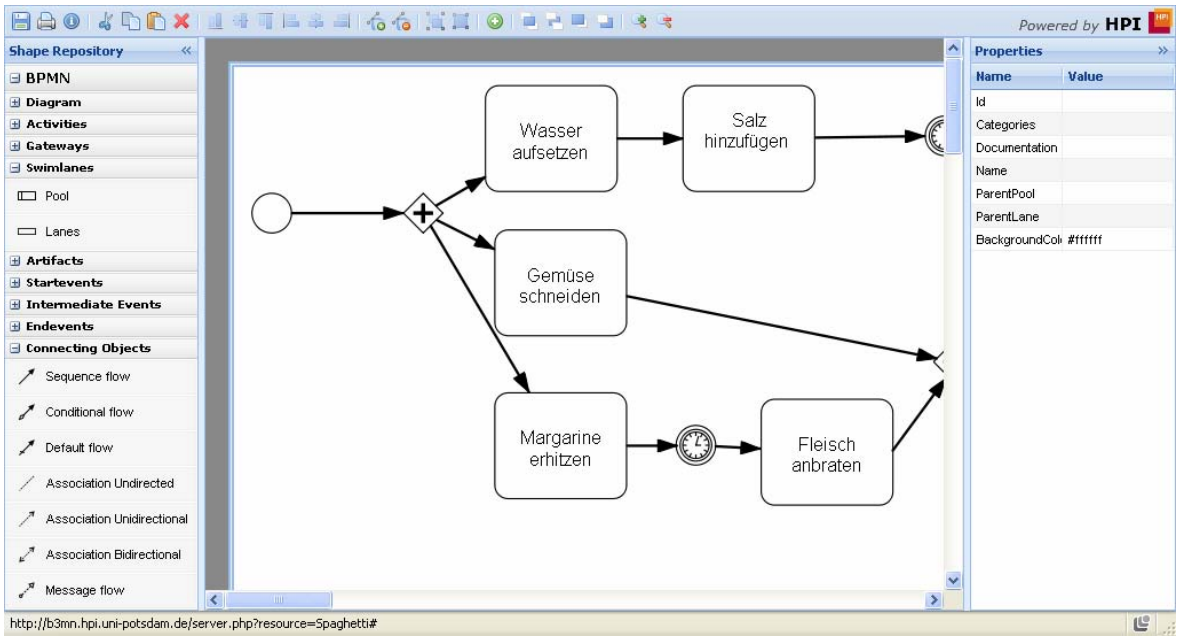


Abbildung 17: Screenshot Oryx

[15] Vgl. http://kgionline.com/xflow2/doc/xflow2/src_as_html/xflow/server/case_condition/case_condition.xf.xml.html

[16] Vgl. C. Ouyang, W.M.P. van der Aalst, M. Dumas and A.H.M. ter Hofstede. Translating BPMN to BPEL, BPM Center Report BPM-06-02, BPMcenter.org, 2006

2.6. Vergleich von Workflow- und Projektmanagementsystemen

Aus der durchgeführten Evaluierung ist ersichtlich geworden, dass eine ganze Reihe an Workflow- und Projektmanagementsystemen mit teilweise unterschiedlichen Funktionalitäten, Bedienkonzepten und Anwendungsdomains existiert, die jedoch auch viele Gemeinsamkeiten zeigen und sich mitunter gegenseitig ergänzen. Diese Gemeinsamkeiten und Unterschiede sollen im Folgenden herausgestellt und zusammengefasst werden, um darauf aufbauend ein semantikbasiertes System zu implementieren, welches einen ähnlichen Funktionsumfang und ein vergleichbares Bedienkonzept bietet.

Zu beachten ist dabei zunächst die Zielsetzung von Workflow- und Projektmanagementsystemen. Workflowmanagementsysteme beschränken sich auf die Modellierung einer Abfolge von Tätigkeiten (Prozessen) und beantworten die Frage, *wer was wann und wie* zu tun hat. Im Mittelpunkt steht die Definition verschiedenster Beziehungen und Abhängigkeiten zwischen einzelnen Aktivitäten, die von unterschiedlichen Beteiligten auszuführen sind, wobei während der Ausführung des Workflows der Einzelne in der Regel nur die durch ihn aktuell durchzuführende Aufgabe sieht und die Einordnung in den Gesamtkontext des Workflows verborgen bleibt. Im Gegensatz dazu finden sich in einem Projektmanagementsystem zwar auch Aufgabenlisten, denen Verantwortliche zugeordnet werden können, jedoch bestehen dort in der Regel keine Abhängigkeiten zwischen diesen, sodass sich meist nur tabellarische Auflistungen finden. Eine größere Priorität erhält in einem Projektmanagementsystem das Ressourcenmanagement im Bezug auf die Fertigstellung einer Aufgabe oder eines Projektteils sowie der damit verbundenen Dokumente und anderer physischer Produkte. Während Workflows sehr feingranular sein können („Arbeitsanweisungen“), wird in Projektmanagementsystemen eine durchzuführende Aufgabe meist nur stichpunktartig in textueller Form beschrieben; die praktische Ausgestaltung ist dem Bearbeiter überlassen.

Die Grundlage von sowohl Workflow- als auch Projektmanagementsystemen ist eine rollenbasierte Nutzerverwaltung, anhand derer zum Einen die Rechte des Benutzers in der Systemumgebung als auch die Zuständigkeiten in einem bestimmten Prozess ermittelt werden. Systemrolle als auch die Rolle in einem konkreten Projekt können bezüglich eines Benutzers unabhängig voneinander festgelegt werden oder auch identisch sein. Im einfachsten Fall gibt es nur eine Rolle, bei der alle Systembenutzer alles dürfen. Es findet sich jedoch kein Projekt- oder Workflowmanagementsystem, welches ohne Nutzerverwaltung oder Zugriffsschutz auskommt, was mit der Sensibilität der Daten als auch mit administrativen Funktionen (Logging der Systemaktivitäten, Benutzerverwaltung) begründet werden kann.

Nach Anmeldung am System und der Feststellung der Rollenzuordnung findet sich bei vielen der getesteten Anwendungen eine Übersichtsseite, welche individuell die Projekte und Aufgaben zusammenfasst, an welchen der Benutzer beteiligt ist und in welchen er momentan eine bestimmte Aufgabe durchzuführen hat. Die weitere Navigation unterscheidet sich mitunter stark zwischen den zur Verfügung stehenden Systemen, eine Navigationsleiste mit einem Schnellzugriff auf verschiedene Module findet sich aber bei den meisten. Das Anlegen und die Verwaltung neuer Systemobjekte (im Speziellen Projekte und darin enthaltenen Aufgaben) unterscheidet sich in der Vorgehensweise zwischen Projekt- und Workflowmanagementsystemen. Objekte in Projektmanagementsystemen können jederzeit manipuliert werden, indem durch Auswahl eines Bearbeiten-Dialogs die Eigenschaftswerte eines Systemobjekts aktualisiert werden können. Dies wird bewusst als Kernkonzept in Projektmanagementsystemen genutzt, um beispielsweise den aktuellen Status und Arbeitsfortschritt eines Projektes jederzeit durch die Beteiligten anpassen zu können. Objekten in Workflowmanagementsystemen können zur Designzeit des Workflows zwar auch entsprechende Attribute wie eine Priorität zugeordnet werden, die ein Workflowobjekt und dessen Bedeutung näher beschreiben, allerdings obliegt die Ausführung des Workflows einer separaten Workflowengine, wodurch ein Prozessbeteiligter nach der Ausführung eines neuen Prozesses nur die workflowspezifischen Daten angeben kann, welche in der Rahmenumgebung der Workflowengine abgefragt werden. Die Überwachung und Verwaltung des Prozessstatus wird hier also an das System abgegeben. Wie bereits in Kapitel 2.2. vermutet, fand sich in allen getesteten Systemen ein ähnliches Begriffsvokabular wieder, was nicht nur auf die Verwendung der Kernelemente wie Projekte, Prozesse oder Aktivitäten beschränkt war, sondern sich auch in den abgefragten Attributen der Eigenschaftsdialoge widerspiegelte. Tabelle 3 und 4 listen die verwendeten Bezeichnungen für einige Objektattribute in den getesteten Systemen auf:

PHProjekt	Double Choco Latte	dotproject	WebCollab	PHPCollab	Basecamp
Name Beginn Ende Priorität Kategorie Budget Bemerkung Leiter Stundensatz Unterprojekt von Status Teilnehmer Kontakte Gruppenfreigabe	Name Projektleiter Deadline Vaterprojekt Beschreibung (ETC) (Angelegt durch) (Status) (Letzte Änderung) (Geschlossen am) (Aufgaben gesamt) (Aufgaben beendet) (Geplante Stunden) (Aktuelle Stunden) (Verbleibende Stunden) (Prozent fertiggestellt)	Name Priorität Besitzer Unternehmen Startdatum Zielenddatum Zielbudget (Berechnetes Enddatum) Derzeitiges Budget URL EntwicklungsURL Status Fortschritt Kurzname Projektfarbe Typ Beschreibung (Gearbeitete Stunden) (Geplante Stunden) (Projektstunden)	Erstellzeit Name Deadline Priorität Status Besitzer Gruppe Öffentlich einsehbar? Öffentlich bearbeitbar? Beschreibung	Name Priorität Beschreibung URL Entwicklungs-URL Besitzer Organisation Phasen? Status Dateigröße Rechnungen? Stundensatz (Aktive Phasen) (Angelegt am) (Geändert am) (Verzeichnisgröße) (Geschätzte Zeit) (Tatsächliche Zeit) (Differenz)	Projektname Beschreibungstext Startseite
JaWE XPDL Editor	Bonita	Imixs	OSWorkflow	Runa	Oryx
Name	Name Beschreibung Version Autor Status		Name Erstellungsdatum		Name Version Autor Sprache Erstellungsdatum Änderungsdatum Beschreibung

Tabelle 3: Typische Objekteigenschaften von Projekten

PHProjekt	Double Choco Latte	dotproject	WebCollab	PHPCollab	Basecamp
Titel Von An Priorität Datum Beginn Termin Projekt Beschreibung Status Arbeitszeit Gruppenfreigabe	Produkt Modul Öffentlich? Typ Kategorie Verantwortlicher Deadline Geplanter Start Geplantes Ende Geplante Stunden Priorität Ernstheit Zusammenfassung Notizen Beschreibung Projekt	Name Status Priorität Fortschritt Meilenstein? Besitzer Typ Zugriffstyp URL Vateraufgabe Budget Beschreibung Startdatum Enddatum Erwartete Dauer Zugewiesene Bearbeiter	Projekt Erstellzeit Name Deadline Priorität Status Besitzer Aufgabengruppe Nutzergruppe Öffentlich einsehbar? Öffentlich beschreibbar? Beschreibung	Projekt Phase Organisation Name Beschreibung Besitzer Status Fortschritt Priorität Startdatum Fälligkeit Geschätzte Zeit Tatsächliche Zeit Kommentare Veröffentlicht? Rechnungen? Gearbeitete Stunden (Angelegt am) (Zugewiesen am) (Geändert am)	Listenname Beschreibung Meilensteinbezug? Aufgabenname Verantwortlicher
JaWE XPDL Editor	**Bonita**	**Imixs**	**OSWorkflow**	**Runa**	**Oryx**
Name Durchführer Startmodus Beendemodus Deadline Priorität Limit Icon Beschreibung	Name Typ Beschreibung Durchführer Deadline		Name		Kategorie Name Aktivitätstyp Implementierungsart Ausführer Inputs Outputs InMessage OutMessage Looptype Loopcounter

Tabelle 4: Typische Objekteigenschaften von Aufgaben

Projekte und Aufgaben bilden die zentralen Elemente, die sich in allen getesteten Managementanwendungen wieder finden. Eine dritte Entität stellen Nutzer und deren Rollenzugehörigkeit dar. Primär Projektmanagement-spezifisch sind darüber hinaus die Zuordnung von Dokumenten. In verschiedenen Produkten finden sich weitere Objekte wie beispielsweise Projektphasen, Notizen oder Terminereignisse, welche in erster Linie jedoch eine Erweiterung der drei Kernobjekte darstellen (so lassen sich Projektphasen beispielsweise auch als Subprojekte abbilden). Alle Systemobjekte besitzen charakteristische Eigenschaften, welche als Attribut-Wert-Paar aufgefasst werden können und sich daher hervorragend für eine semantische Interpretation eignen.

Abschließend kann festgestellt werden, dass sich Projekt- und Workflowmanagementsysteme prinzipiell gegenseitig ergänzen könnten. Zum einen könnte mithilfe von Workflowgraphen die Durchführung von Aufgaben innerhalb eines Projektes vorab modelliert und die Realisierung eines Projektes mithilfe von Prozessbeschreibungen dargestellt und überwacht werden, wodurch de bisher sehr auf manuelle Eingaben eines Managers angewiesene Bereich des Projektmanagements durch die Praktiken aus dem Workflowmanagement profitieren könnte. Zum anderen fehlen klassischen Workflowmanagementanwendungen projekttypische Funktionalitäten wie die Einordnung in einen Gesamtkontext oder die Dokumentenverwaltung, was zwar durch Verwendung externer Systeme problemlos ergänzbar ist, für kleinere Prozesse eine integrierte Lösung aber häufig geeigneter wäre.

Diese Idee konnte ansatzweise während der Evaluierung bei einigen Systemen bereits dahingehend festgestellt werden, dass es zu Inkonsistenzen in der Begriffsverwendung kam. So wurde das Anlegen eines neuen Workflowgraphen bereits als Projekt bezeichnet, oder in Projektmanagementsystemen wurden Möglichkeiten angeboten, Projekte und Aufgaben hierarchisch zu strukturieren oder zu reorganisieren. Durch eine konsistentere Zuordnung von Eigenschaften zu einem bestimmten Objekt könnte eine nahtlose Integration beider Managementkategorien hergestellt werden (bspw. braucht eine Aktivität/Aufgabe ein definiertes Start- und Enddatum?). Im nachfolgenden Kapitel soll nun diskutiert werden, wie diese Erkenntnisse in ein semantikbasiertes System transformiert werden können, welches sich vor allem auf ein eindeutiges Grundvokabular stützt.

3. Konzeption eines semantikbasierten Projektmanagement-systems

„The Internet must be the most difficult environment conceivable for search engines. It's huge, growing, always changing, and inconsistent, and it contains documents of all kinds and structures. These documents may contradict each other, and no central registry lists them all." ~ Explorer's Guide to the Semantic Web, Thomas B. Passin

3.1. Was bedeutet Semantik?

Soll im Weiteren ein semantikbasiertes Projektmanagementsystem mit den Grundvorstellungen des Semantic Web entworfen werden, so steht am Beginn zunächst die Frage, was unter dem Begriff "Semantik" eigentlich zu verstehen ist. Für einen menschlichen Benutzer ist die Bedeutung eines Konzeptes intuitiv erfassbar. So können strukturelle Informationen wie in der Zeichenkette „*01.11.2007*" direkt dazu benutzt werden, abzuleiten, dass damit wahrscheinlich eine Datumsangabe ausgedrückt wird. Die Zeichenfolge „*Diplomarbeit*" innerhalb eines Textes wird direkt mit einer wissenschaftlichen Ausarbeitung assoziiert, welche meist *um die 100 Seiten* umfasst und sich *mit einer Thematik kontrovers auseinandersetzt*. Automatisch werden damit weitere Konzepte der realen Welt assoziiert, sodass die Begriffe „*Paper*" oder „*Thesis*" in eine ähnliche Relation zu dem Begriff „*Ausarbeitung*" gesetzt werden können und entsprechend gleiche oder ähnliche Eigenschaften zugeschrieben bekommen, welche von der abstrakten Gesamtgruppe bekannt sind, und entsprechend dann auch im gleichen Kontext verwendet werden können. Warum ist diese Vorgehensweise nicht direkt auf eine rechnergestützte Verarbeitung übertragbar? Ein Computerprogramm ist prinzipiell genauso in der Lage, einzelne Buchstaben als zusammenhängende Zeichenkette zu erkennen und bestimmte Anforderungen an die Struktur und das erlaubte Verwendungsumfeld zu stellen (vgl. Datentyp). So kann ein Typ *date* definiert werden, der nur aus Ziffern zwischen 0-9 sowie zwei Punkten nach dem zweiten und vierten Zeichen besteht, wobei die Gesamtlänge der Zeichenkette 10 Zeichen nicht übersteigen darf und weitere Bedingungen an die erlaubten Zahlenwerte je Position existieren. Wird dieses Muster beispielsweise auf einer Webseite gefunden, könnte eine Applikation diese Datumsangaben auslesen, weiterverarbeiten oder für einen Export bereitstellen.[17] Eine alternative Möglichkeit wäre, ein Datum explizit auszuzeichnen (bspw. <*date*>*01.11.2007*</*date*>). Das Problem besteht darin, dies auf alle anderen Konzepte der realen Welt zu übertragen, wie das Diplomarbeits-Beispiel recht gut darstellt.

[17] Praktisch wird dies beispielsweise durch das Skype -Browser-Plugin realisiert, wo Webseiten syntaktisch nach enthaltenen Telefonnummern analysiert werden

Ohne große Probleme könnte ein XML Schema entworfen werden, welches syntaktisch korrekt die Bestandteile eines einzelnen Objektes sowie gültige Eigenschaftswerte definiert und einzuschränken vermag. Erfüllen die angegebenen Daten in einer vorhandenen XML-Datei diese Schemadefinition, so werden diese unter der durch den Programmierer angegebenen Struktur weiterverarbeitet. Ob der angegebene Inhalt einen „Sinn“ ergibt, ist dabei der Deutung durch den menschlichen Benutzer überlassen. Die Ursache dafür ist, dass eine Computeranwendung nur vorhandene Daten verarbeiten kann, der Transformationsschritt der Daten in eine Information jedoch ohne weiteres nicht durchführbar ist.

Definition 7: Syntax
Syntax ist die Lehre vom strukturellen Satzbau und umfasst ein System an Regeln, wie aus einer Menge von Grundsymbolen durch spezifische syntaktische Mittel gültige und wohlgeformte Worte, Wortgruppen und Sätze einer Sprache abgeleitet werden können.

Definition 8: Semantik
Semantik ist die Lehre von der inhaltlichen Bedeutung sprachlicher Ausdrücke, sowie deren Beziehungen untereinander.

Definition 9: Kontext
Kontext kennzeichnet den Zusammenhang oder die Bedingung, unter dem eine Aussage Gültigkeit erlangt.

Daten werden nach definierten Syntaxregeln aus Zeichen eines bestimmten Zeichenvorrats gebildet. Informationen werden aus Daten unter einer bestimmten Bedeutungszuordnung gewonnen. Die Bedeutung der Daten ist dabei von einem verwendeten Kontext abhängig. Informationen können anschließend als eine Repräsentationsform für Konzepte der realen Welt verwendet werden. Abbildung 18 verdeutlicht diese Begriffshierarchie, wobei die suggerierte scharfe Trennung zwischen den Begriffswelten nach Bodendorf in der Realwelt nicht immer zu finden, sondern eher als kontinuierlicher Übergang zu verstehen ist [Bod03, p.2f].

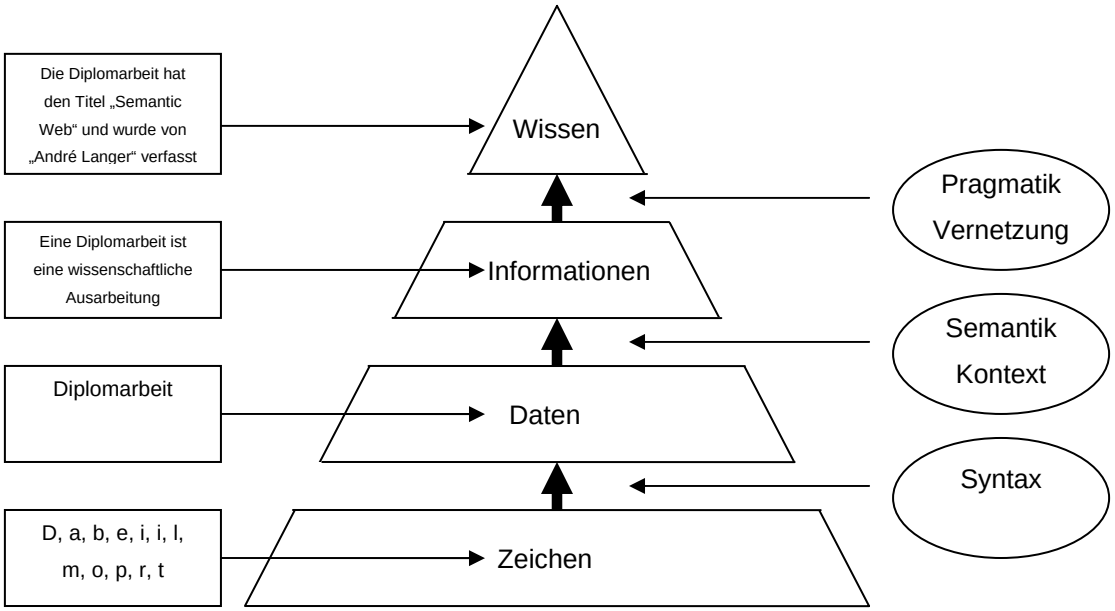


Abbildung 18: Wissenspyramide nach Bodendorf [Bod03]

Die Herausforderung ist es also, syntaktisch korrekte Daten so mit weiteren Angaben zur Bedeutung der Daten zu versehen, dass die daraus abgeleitete Information rechnergestützt weiterverarbeitet werden kann. Interessant ist dabei die Grundannahme, dass dies ohne eine direkte Beteiligung einer „intelligenten“ Prozessorinstanz vonstatten gehen kann, was in direktem Widerspruch zu dem Ansatz aus dem Bereich der künstlichen Intelligenz steht, welche die Entwicklung intelligenter Agenten anstrebt. Praktisch bedeutet dies das Ziel, Informationen zueinander in Beziehung setzen zu können, semantische Widersprüche auflösen zu können und aus den zugrunde liegenden Daten Informationen so extrahieren zu können, dass sie einer bestimmten Fragestellung genügen und diese semantisch korrekt beantworten. Ein Computerprogramm muss dazu nicht wissen, wie ein „*Diplomarbeit*“ in der Realwelt aussieht, sondern muss in die Lage versetzt werden, hierarchische Beziehungen zwischen verschiedenen Konzepten herstellen zu können und in der Lage sein zu entscheiden, ob eine konkrete Information einem Konzept angehört oder nicht.

Gelingt dies, so wird entsprechend Abbildung 16 dadurch der Schritt in eine darüber liegende Ebene des Wissens möglich.

Definition 10: Wissen (nach Haun [Hau02])
„Wissen kann beschrieben werden als in einen bestimmten Kontext gestellte Information, die für denjenigen, der über diese Information verfügt, von Wert ist und ihn dazu befähigt, etwas zu tun, wozu er ohne dieses Wissen nicht in der Lage gewesen wäre."

Wissen entsteht durch dabei durch die Verknüpfung und Interpretation von Informationen. Ohne Einordnung in einen bestimmten Problemkontext bzw. eine bestimmte Fragestellung, welche durch ein Individuum unter Rückgriff auf eine bestimmte Wissensbasis versucht wird zu beantworten, ist der Wissensbegriff inhaltsleer. Übertragen auf das Internet mit den darin enthaltenen Millionen von Daten und daraus ableitbaren Informationen stellt dies die Motivation dar, warum eine Vernetzung und domainübergreifende Auswertung der Daten in einem Semantic Web weitreichende positive Ergebnisse hätte. Der Bereich des Wissensmanagements beschäftigt sich darüber hinausgehend mit vielen weiteren Fragestellungen, inwieweit Wissen veraltern, neu generiert oder verfälscht wird kann, was an dieser Stelle aber nicht weiter vertieft werden soll. Wichtig ist, dass allein durch die Vernetzung verschiedener Aussagen, die in unterschiedlichen Ressourcen enthalten sind, neue Informationen so abgeleitet und aggregiert werden können, dass sie dem Fragesteller eine zweckgerichtete Antwort auf eine konkrete Problemstellung geben können (Pragmatik). Diese Ableitung aus partiell vorhandenem Wissen wiederum erfordert kein vollständiges Weltbild, sondern lediglich ein Wissen über bestimmte Zusammenhänge und kann strukturell abgebildet werden, wodurch bisher individuell durchgeführte Suchaufträge automatisiert werden können.

3.2. Wissensbeschreibung

Bevor Techniken zur informationstechnischen Beschreibung der Bedeutung von Daten vorgestellt werden können, ist noch eine prinzipielle Fragestellung zu klären, wie Bedeutungen generell beschrieben werden können, nachdem in Kapitel 3.1. bereits gezeigt wurde, dass zu einer Wissensanalyse der Rückgriff auf grundlegende Zusammenhänge zwischen Konzepten der realen Welt ausreicht. Damit diese allgemeingültig verstanden, aber auch zwischen verschiedenen Individuen ausgetauscht werden können („Begriffsverständigung"), definiert Bodendorf drei wesentliche Anforderungen an eine Wissensbeschreibung zur Auflösung semantischer Konflikte und zur Sicherung von Wiederverwendbarkeit des Wissens [Bod03, p.107ff]:

- **Symbole**: Es muss zur Referenzierung das gleiche Begriffssymbol verwendet werden (es wird von einem Objekt mit der Bezeichnung „*Diplomarbeit*" gesprochen) → Terminologie nötig
- **Zuordnungen**: Unter einem konkreten Kontext muss das Begriffssymbol von jeder Verarbeitungsinstanz den gleichen Konzepten zuordbar sein (eine „*Diplomarbeit*" ist eine „*wissenschaftliche Ausarbeitung*") → semantische Schemata nötig
- **Konzepte**: Jede der Verarbeitungsinstanzen muss einem gewissen Konzept die gleiche Bedeutung zumessen (*Eine wissenschaftliche Ausarbeitung umfasst mehere hundert Seiten und beschäftigt sich mit einer konkreten Fragestellung*.)
  → Ontologien nötig

Dies bedeutet jedoch nicht, dass keine Umbenennungsoperationen existieren können (bspw. Eine „*Diplomarbeit*“ ist eine „*Abschlussarbeit*“). Es wird lediglich die Forderung aufgestellt, dass der Verarbeitungsinstanz zum jeweiligen Analysezeitpunkt die verwendeten Symbole des Ursprungssystems bekannt sein müssen. Wurde Wissen durch Verwendung von Symbolen formuliert, deren Bedeutung zum aktuellen Zeitpunkt nicht hergeleitet werden kann, so kann dieses Wissen nicht benutzt werden.

Um eine Verständigung auf symbolischer Ebene erreichen zu können, werden von Anwendungen Terminologien benutzt.

Definition 11: Terminologie
Eine Terminologie ist die Gesamtheit aller gültigen Bezeichnungen in einer Anwendungsdomain.

Mit Terminologien können beispielsweise alle gültigen Eigenschaftswerte aufgelistet werden, die eine konkrete Objekteigenschaft annehmen kann. Da diese Auflistung applikationsabhängig lokal innerhalb der Systemgrenzen definiert ist, birgt diese eine Gefahr bei zukünftiger Anpassung oder Änderung der Symbolmenge. Wird beispielsweise für gültige Werte eines Projektstatus die Menge *{nicht begonnen, laufend, abgeschlossen}* definiert und später der Wert *{abgebrochen}* hinzugefügt, so wird dieser von Anwendungen, welche die alte Terminologie verwenden, nicht verstanden. Eine Terminologie definiert damit eine syntaktische Norm auf Symbolebene, der semantische Aspekt hingegen wird mithilfe weitreichenderer Repräsentationen dargestellt.

Definition 12: Semantisches Schema (nach Bodendorf [Bod03, p.109])
Ein semantisches Schema stellt die systemweit einheitliche Zuordnung von Symbolen zu Konzepten sicher, indem eine Menge von Konzepten definiert und aufgezählt wird.

Semantische Schemata dienen der Zuordnung von Begriffen zu Konzepten, wobei damit auch unterschiedliche Bezeichnungen, die das gleiche Objekt repräsentieren, in das gleiche Konzept eingeordnet werden können. Praktisch kann dies so aussehen, dass einem Konzept *dc:language* beispielsweise die Symbole *„german“, „english“, „ger“, „en“, „de“* oder *„deutsch“* zugeordnet werden können. Dadurch wird eine bedeutende Interoperabilität in verteilten Systemen ermöglicht, da diese eine Norm auf Zuordnungsebene definieren. Ohne eine entsprechende Zuordnungsvorschrift könnte ein beliebiges System das vorangestellte Konzept ebenso als *dc:lang, dc:sprache* oder *dc:verfasstin* kennzeichnen, wodurch kein Gewinn auf Bedeutungsebene im Gegensatz zur Symboldefinition zu erkennen wäre. Symbole erhalten durch die Zuordnung eine semantische Identität, indem sie in Mengen mit gleichen Grundbedeutungen gruppiert werden.

Um nun auf der obersten Abstraktionsebene der Wissensbeschreibung eine Aussage über die Bedeutung der Konzepte treffen zu können, werden Ontologien verwendet. Die Bezeichnung Ontologie stammt aus dem Griechischen (*ontos*-Lebenwesen, *logos*-Wort) und wurde ursprünglich in der Philosophie des Neunzehnten Jahrhunderts von Rudolf Gockel dazu verwendet, die Untersuchung des Wesens von existierenden Objekten von der biologischen Untersuchung von Lebewesen abgrenzen zu können. Im Mittelpunkt steht die Erforschung von Objektkategorisierungen einer bestimmten Domain, primäres Hilfsmittel dazu „is *a catalogue of the types of things that are assumed to exist in a domain of interest D from the perspective of a person who uses a language L for the purpose of talking about D*“ [Sow03]. Im Grunde genommen verbirgt sich dahinter die Klassifizierung von Objekten einer bestimmten Umgebung sowie deren Beziehungen untereinander. Im Gegensatz zu einer Taxonomie sind diese Beziehungen jedoch nicht auf eine streng hierarchische Struktur begrenzt.

Definition 13: Ontologie (frei übersetzt nach Gruber [Gru93])
Eine Ontologie ist eine formale, explizite Spezifikation eines gemeinsam verwendeten, abstrakten Modells zur Repräsentation des darin enthaltenen Wissens.

Eine Ontologie leistet damit zwei Dinge: Zum Einen wird explizit ein Vokabular (d.h. eine Liste an Begriffen) bereitgestellt, welches systemübergreifend verstanden und benutzt werden und damit eine Kommunikation ohne Mehrdeutigkeiten zwischen diesen Systemen ermöglicht wird. Zum anderen wird die Bedeutung eines Konzeptes eingegrenzt, indem typische Eigenschaften der zugeordneten Objektinstanzen des Konzeptes benannt und zu anderen Konzepten in Beziehung gesetzt werden. Diese Beschreibung basierend auf Beziehungen zwischen verschiedenen Konzepten kann unterschiedlich komplex sein und sich je nach Anwendungsfall einer informellen Komplettbeschreibung mehr oder weniger stark annähren, wodurch die Bedeutung eines Konzeptes mit Schwerpunkt auf die wesentlichen, innerhalb einer Anwendungsdomain zu betrachtenden Aspekte, eindeutig und maschinell verarbeitbar dargestellt werden kann. Die durch ein semantisches Schema *definierten* Konzepte können unter Zuhilfenahme einer Ontologie miteinander verknüpft werden, wobei dabei eine Computeranwendung in die Lage versetzt wird, unter Zuhilfenahme der Konzeptdefinition innerhalb einer Ontologie festzustellen, ob eine gewisse Beziehung zwischen zwei Konzepten gültig ist und welche Bedeutung dieser zukommt.

Ontologien werden dazu unter dem Anspruch entworfen, innerhalb einer bestimmten Problemdomain unabhängig von einem konkreten Anwendungsfall nutzbar zu sein und darin vorhandene Konzepte allgemeingültig zu beschreiben. Die Idee dahinter ist, bereits vorhandene Ontologien für ähnliche Problemstellungen weiternutzen zu können.

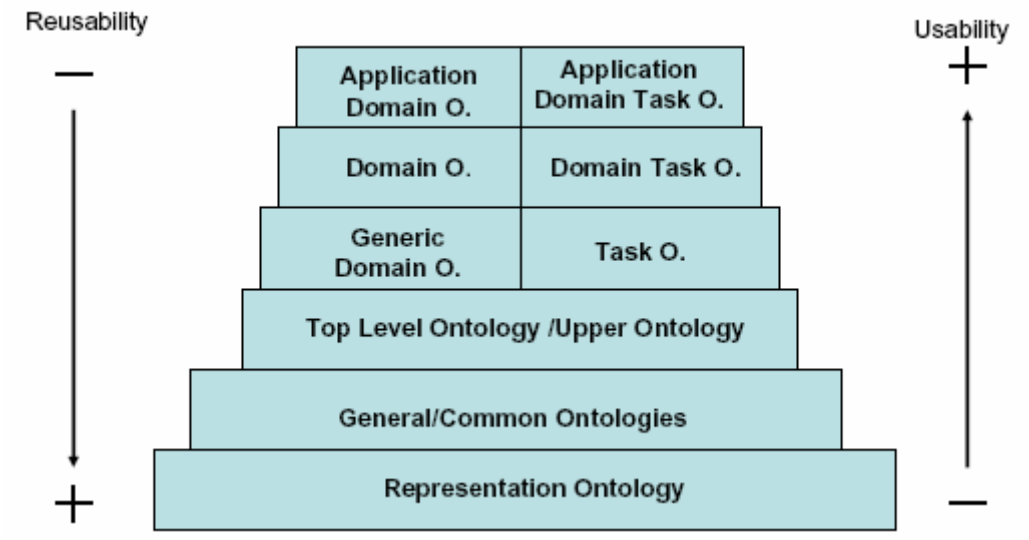


Abbildung 19: Reusability-usability trade-off Problem, Quelle: http://buell.izbi.uni-leipzig.de/gridworkshop/presentations/Weller_Ontologien%20im%20Wissensmanagement.pdf

Die Feingranularität einer Ontologiebeschreibung kann dabei stark variieren; es darf nur nicht möglich, unter der Verwendung einer Ontologie eine Aussage zu formulieren, welche dem durch die Ontologie modellierten Weltbild direkt widerspricht. So entstanden neben einfachen bis komplexen *Domain-spezifischen* Ontologien ebenso Versuche, generalistische Ontologien zu entwickeln, welche den Anspruch erheben, eine allumfassende Weltbeschreibung (*Top Level Ontologies*) bereitzustellen, vgl. Klassifikation nach Guarino [Gua98]. Bekannte Vertreter davon sind die Suggested Upper Merged Ontology (SUMO) sowie die CYC-Ontologie. Abbildung 19 illustriert das Trade-Off-Problem zwischen Verwendbarkeit und Wiederverwendbarkeit von Ontologien, weswegen die Entwicklung von Top-level/Upper Ontologies von vielen Forschern momentan eher der Grundlagenforschung zugeschrieben wird, und sich um den Aufbau eines allumfassenden Begriffkataloges bemüht.

3.3. Überblick über Technologien zur Repräsentation von Semantik

3.3.1. Motivation

In Kapitel 3.1. wurde gezeigt, dass eine Bereitstellung maschinell auswertbarer semantischer Meta-Daten im Internet durch eine beliebige Umsetzung in XML nicht realisiert werden kann. Dies bedeutet nicht, dass XML dazu generell nicht geeignet wäre, da gerade die Flexibilität des Austauschformats sowie die Erfahrungen und entwickelten Werkzeuge der letzten Jahre dafür genügend Freiraum lassen. Vielmehr ist eine strukturelle Einschränkung und Definition fester Schlüsselwörter mit einer grundlegenden Bedeutung nötig, die von einem Prozessor ohne vorhandene Mehrdeutigkeiten verarbeitet werden können.

In den zurückliegenden Jahren wurden eine Reihe von Sprachen entwickelt mit deren Hilfe Ontologien implementiert werden können, die entweder in XML oder einem anderen Format abbildbar sind. Dieser Abschnitt soll dazu dienen, die einzelnen Sprachkonzepte gegenüber zu stellen und daraus resultierend ein geeignetes Ausdrucksmittel zur Implementierung einer Projektmanagement-Ontologie zu wählen. Es findet dabei eine Beschränkung auf *web-based ontology languages* statt, welche direkt zum Mark-up von Ontologien verwendet werden können. Daneben existieren eine Reihe weiterer Entwicklungen, die dem Bereich der künstlichen Intelligenz zu Beginn der Neunziger Jahre entstammen, auf die an dieser Stelle jedoch nicht vertieft eingegangen werden soll. Dazu zählen in erster Linie das Knowledge Interchange Format (KIF) mit einer abstrakteren Verwendung in Ontolingua[18], sowie die Sprachen LOOM[19], die Operational Conceptual Modelling Language (OCML)[20] und F-Logic, welche alle auf der Prädikatenlogik 1. Ordnung basieren und einen Lisp oder Prolog-ähnlichen Syntax aufweisen.

3.3.2. RDF

Das Konzept, Metadaten zur Beschreibung und zum Auffinden von Inhalten zu benutzen, wird im Alltag (bspw. Bibliotheksbereich) schon seit längerer Zeit genutzt. Mit der Grundidee des heutigen Internets, ein Informationsmedium an der Schnittstelle zwischen Mensch und Maschine zu schaffen, wurde die systematische Verwaltung von Metadaten zur Beschreibung von Internetinhalten bereits zu Beginn als essentiell wichtig angesehen und nach entsprechenden Lösungsansätzen gesucht, um beispielsweise das Erstellungsdatum, den Autor oder den Titel eines Dokumentes im Internet explizit auszeichnen zu können. Ende 1996 wurden dazu verschiedene Vorschläge basierend auf XML bei dem World Wide Web Consortium (W3C) eingereicht, die in der Flexibilität dieses Formats zur Auszeichnung beliebiger Informationen eine Anwendungsmöglichkeit zur Darstellung von Metadaten sahen[21]. Sowohl von Microsoft (XML-Data basierend auf Web Collections sowie dem Channel Definition Format (CDF)) als auch von Netscape (Meta Content Framework (MCF)) wurden zu dieser Zeit entsprechende Vorschläge unterbreitet, die von einer Arbeitsgruppe des W3C zu einem Ressource Description Format weiterentwickelt worden, wobei das Meta Content Framework von Ramanathan Guha[22] (Netscape) wesentlichen Einfluss auf den ersten Entwurf des Ressource Description Frameworks aus dem Oktober 1997 hatte. Seit 10. Februar 2004 ist RDF in einer W3C Recommendation offiziell spezifiziert[23].

[18] Vgl. http://www.ksl.stanford.edu/software/ontolingua/

[19] Vgl. http://www.isi.edu/isd/LOOM/

[20] Vgl. http://kmi.open.ac.uk/projects/ocml/

[21] Vertiefend sei auf http://www.heise.de/ix/artikel/1998/01/116/ verwiesen

[22] Siehe http://www.w3.org/TR/NOTE-MCF-XML/

[23] Vgl. http://www.w3.org/2001/sw/#spec

Grundgedanke ist, dass sich jede Aussage über eine Ressource als Tupel aus drei Informationen darstellen lässt. Eine Ressource ist dabei ein beliebiges, eindeutig identifizierbares (bezeichenbares) Objekt. Dies können physisch vorhandene Dokumente im Internet sein, welche direkt adressiert werden können, aber auch abstrakte, symbolisch benannte Objekte, welche ein Konzept der Realwelt kennzeichnen. Die Bezeichnung einer Ressource geschieht durch die Verwendung von Uniform Ressource Identifiers (URIs). Das als RDF statement bezeichnete Tripel *(S,P,O)* besteht aus der Angabe eines Subjekts *S* (über welches eine Aussage gemacht wird), eines Prädikats *P* (manchmal auch als *property* bezeichnet, welches die binäre Relation charakterisiert) und eines Objekts *O*, wobei die Objektinformation entweder literal sein, aber auch auf eine weitere Ressource verweisen kann. Durch diese Herangehensweise entsteht eine Graphstruktur, welche Aussagen beliebiger Komplexität zulässt. Durch die Verwendung eindeutiger Bezeichner für referenzierte Objekte können Mehrdeutigkeiten aufgelöst werden (identische Objekte haben die gleiche URI), wodurch die Verarbeitung von RDF-Informationen zu einer Graph matching – Aufgabe wird (Isomorphismus). Die RDF-Spezifikation umfasst dabei sowohl das zugrunde liegende Datenmodell zur Darstellung als XML-Graph, als auch eine Syntax-Spezifikation zum Austausch von RDF-Beschreibung in RDF/XML.

3.3.3. N-Triples

Um ein RDF-Modell zu serialisieren, um es speichern und austauschen zu können, ist RDF/XML nicht die einzige Möglichkeit. Wesentlicher Vorteil des RDF/XML-Serialisierungsformats ist, dass es direkt in XML abgebildet werden kann und sich damit hervorragend zur Eingliederung in bestehende XML-Dokumente im Web eignet. Daneben existiert jedoch der Nachteil, dass RDF/XML-Strukturen schnell sehr komplex und schwer fassbar werden können, da darin eine Baumstruktur abgebildet wird und es schwierig werden kann, die ursprünglichen RDF-Tripel darin auf einen Blick zu identifizieren. Dies kann besonders dann zu einem Problem werden, wenn die Ergebnisse einer auf RDF-basierenden Operation mit erwarteten Ergebnissen durch Anwendung einfacher Textvergleichsoperationen verglichen werden sollen, da die darunter liegenden Graphmodelle unterschiedlich dargestellt werden können, obwohl sie die gleiche Struktur und Information enthalten. Ein Lösungsansatz wäre, alle Modelle nach einer durchgeführten Operation zu normalisieren, um sie mit einer Art Referenzmodell eindeutig vergleichen zu können. Das W3C verfolgte einen alternativen Weg und entwickelte speziell zum Einsatz in Testumgebungen zur Überprüfung von Testfällen ein separates, sehr einfach gehaltenes Textformat unter der Bezeichnung N-Triples. Im Unterschied zu RDF/XML ist N-Triples ein zeilenbasiertes Format, worin alle RDF-Tripel auf jeweils einer separaten Zeile als leerzeichengetrennte Aufzählung von Subjekt, Prädikat und Objekt dargestellt werden, wobei alle Ressourcen und Prädikate durch ihre vollständigen URI aufgeführt werden. Weitere reservierte Zeichen und syntaktische Regeln existieren, um mehrere Aussagen über ein Subjekt zusammenzufassen, oder um über die eigentliche Serialisierung des RDF-Modells hinaus zusätzliche Informationen wie Inferenzregeln einzubetten.

N-Triples stellt damit ein einfaches Textformat dar, welches gleichzeitig maschinell verarbeitet als auch direkt von einem menschlichen Akteur gelesen und verstanden werden kann. Im Kern ist es eine Teilmenge der Notation 3 (N3), welche die Einbettung von RDF statements in Nachrichten erlaubt und ursprünglich von Sir Tim-Berners-Lee selbst entwickelt wurde. Eine weitere Variante unter der Bezeichnung Terse RDF Triple Language[24] (Turtle) greift wesentliche Aspekte dieser Sprache auf mit der Zielsetzung, einen korrekten Syntax zur alleinigen Beschreibung von RDF Graphen zu definieren[25].

3.3.4. RDFS

Das Ressource Description Framework (RDF) wird in vielen Veröffentlichungen zu dem Thema bereits als eine leichtgewichtige Sprache zur Beschreibung von Ontologien angesehen. Das dazu in 3.1. definierte Kriterium einer Sicherstellung der Symbolkongruenz kann durch das eindeutige Benennungsschema erfüllt werden, ebenso ist eine Aussage über die Zugehörigkeit einer Ressource zu einem Konzept dahingehend möglich, dass mithilfe eines Prädikats eine Relation zu einer anderen Ressource hergestellt wird. Im engeren Sinne ist dadurch der Klassenbegriff eines Konzeptes der realen Welt aber noch nicht umsetzbar, da bis auf eine *rdf:type*-Eigenschaft keinerlei Gruppierungsmöglichkeiten existieren. Neben der Definition von Konzepten und Relationen sind zur Beschreibung einer Ontologie außerdem Verfahren nötig, gewisse Naturgesetze (Axiome) festzulegen, welche für diese Konzepte allgemeine Gültigkeit haben und durch eine Relation zu einem anderen Objekt nicht zu charakterisieren sind. Daneben sollte es möglich sein, bestimmte Eigenschaftswerte für Konzepte definieren zu können. Eine Erweiterung von RDF wurde nötig, welche die Definition anwendungsspezifischer Klassen und darauf definierter Einschränkungen (constraints) einschließlich weiterer Beziehungen zu anderen Klassen ermöglicht, was zur Entwicklung einer RDF Vocabulary Description Language - RDF Schema (RDFS) führte.

RDFS stellt Möglichkeiten bereit, ein Konzept in Form einer Klasse auszuzeichnen und mithilfe der Eigenschaft *rdfs:subClassOf* eine hierarchische Beziehung zwischen diesen Klassen herzustellen. Darüber hinaus kann die Anwendung beliebig definierter Prädikate mithilfe der Eigenschaften *rdfs:domain* und *rdfs:range* auf zulässige Objektgruppen für Subjekt- und Objektressourcen eingeschränkt werden. Dem Gegenüber werden alle RDF statements, deren *rdf:type*-Eigenschaft auf eine konkrete Klasse verweisen (welche wiederum den *rdf:type*-Eigenschaftswert *Class* besitzt), als ein Individuum (Instanz dieser Klasse) angesehen. Mit diesen Hilfsmitteln können wesentliche Beziehungen zwischen Konzepten als auch die Konzepte selbst beschrieben werden.

[24] Vgl. http://www.dajobe.org/2004/01/turtle/

[25] Verschiedene RDF Serialisierungsformate werden ausführlich unter http://www.dajobe.org/2003/11/new-syntaxes-rdf/ gegenübergestellt

Um darüber hinausgehende, komplexere Beschreibungen von semantischen Zusammenhängen zu ermöglichen, wurden weitere Ontologiebeschreibungssprachen entwickelt, welche auf RDF(S) als Kombination aus RDF und RDFS aufsetzen, da RDF(S) einige Primitive nicht bereitstellt, welche in komplexeren Anwendungen zur Wissensrepräsentation benötigt werden.

3.3.5. DAML+OIL

Wie in Kapitel 1.3. bereits beschrieben, gab es im Jahr 2000 sowohl in Amerika als auch in Europa Bestrebungen, eine allgemeingültige Beschreibungssprache für Inhalte im Semantic Web zu entwickeln. In den USA förderte das Department of Defence die Entwicklung der DARPA Agent Markup Language (DAML), während in Europa im Rahmen des Projektes On-To-Knowledge[26] parallel dazu an einer expliziten Repräsentation von Semantik durch die Ontology Inference Language (OIL, manchmal auch Ontology Interchange oder Integration language) geforscht wurde. Beide Konzepte lieferten wichtige Erkenntnisse für weitere Forschungsarbeiten und wurden schließlich im Dezember 2000 zu DAML+OIL vereint. Grundlage für eine DAML+OIL-Beschreibung einer Ontologie bildet RDF(S), welches um zusätzliche Primitive erweitert wird. Wesentlich dazu ist ein Objekttyp *daml:Ontology*, der definiert, dass es sich bei der vorliegenden Beschreibung um eine Ontologie handelt und weitere Informationen zur Dokumentversion und zu importierenden, externen Dokumenten enthält. *Rdfs:Class* wird durch ein Tag *daml:Class* ersetzt, mit dem eigenständige Klassen definiert werden können. Anstatt darauf nun nur hierarchische Beziehungen (*rdfs:subClassOf*) definieren zu können, sind weitere Prädikate wie beispielsweise *daml:disjointWith,daml:complementOf, daml:intersectionOf* u.a. möglich. Ebenso können Restriktionen auf Prädikate auf einer feineren Ebene modelliert werden, sodass eine *daml:Property* beispielsweise als Identifikationsmerkmal oder als inverse Eigenschaft gekennzeichnet werden kann. Abschließend können verschiedenste Einschränkungen explizit in die Ontologiebeschreibung aufgenommen werden, welche nicht nur zulässige Konzepte sondern auch deren Kardinalität beschränken können.

3.3.6. OWL

Die Web Ontology Language (OWL) ist das Ergebnis einer Arbeitsgruppe (WebOntology Working Group) des W3C, welche sich größtenteils an den Konzepten von DAML+OIL orientierte und diese Sprache im Wesentlichen ersetzte. Seit Februar 2004 existiert auch für die Web Ontology Language eine offizielle W3C Recommendation[27], worin je nach Anwendungsfall drei verschiedene Versionen der Web Ontology Language definiert werden: OWL Lite, OWL DL und OWL Full, die sich in der Ausdruckskraft unterscheiden und deren Befehlssatz nach Aspekten der Berechenbarkeit getrennt ist.

[26] Siehe http://www.ontoknowledge.org/

[27] Vgl. http://www.w3.org/TR/2004/REC-owl-features-20040210/#s1.1

Durch die Ableitung aus DAML+OIL finden sich viele der darin definierten Eigenschaften in OWL wieder, welche in einigen Fällen jedoch neu benannt worden. Die explizite Auszeichnung einer Ontologie mit Meta-Informationen ist erhalten geblieben, ebenso die Erweiterung von RDF(S) mit zusätzlichen Primitiven zur feingranulareren Beschreibung von Objekten und Eigenschaften. Zusätzlich zu den bereits aus DAML+OIL bekannten Eigenschaftseinschränkungen wurde die Möglichkeit ergänzt, symmetrische Eigenschaften zwischen zwei Klassen definieren zu können. Zur Modellierung von Konzepten in Form von komplexen Klassenbeschreibungen stellen die beiden Typen *owl:Thing* als allgemeinste Klasse und *owl:Nothing* als leere Menge die beiden zentralen Konzepte dar, welche das Universum in OWL abgrenzen, von denen ausgehend weitere Klassenbeziehungen modelliert werden können. Über eine ausführlichere Darstellung der Möglichkeiten, mittels OWL eine Ontologie vollwertig zu beschreiben, sei auf die entsprechende Literatur verwiesen.

3.3.7. XML Topic Maps (XTM)

Um Wissen über einen bestimmten Ausschnitt oder Aspekt der Realwelt strukturell abbilden zu können, existieren neben dem RDF-Modell weitere mögliche Herangehensweisen. Eine davon stellen Topic Maps dar, für welche seit 1999 ein ISO-Vorschlag existiert (ISO/IEC 13250), der als aktueller Standard in einer Version von 2003 vorliegt. Informationen werden darin in einer Art Wissensstruktur zunächst ähnlich wie bei einem RDF Graph in Form von Knoten und Kanten angeordnet. Wesentlicher Unterschied ist jedoch, dass zur Erstellung dieser Wissensstruktur eine Anordnung verwendet wird, welche sich stärker an einer natürlichen Wissensverarbeitung durch den Menschen orientiert. Häufig wird zur Umsetzung einer Topic Map der Vergleich mit dem Index eines Buches oder anderen Informationsverzeichnisses herangezogen, der sowohl die Bezeichnung der darin enthaltenen Begriffe („*Topic*“), deren Auftreten im zugrunde liegenden Text („*Occurence*“) als auch Beziehungen zu anderen Themen („*Assocation*“) enthalten kann. Die dadurch abbildbaren Informationen in Topic Maps dienen der Organisation, Strukturierung und Navigation in großen Informationsmengen. Während RDF Aussagen über eine konkrete Ressource macht, eignen sich Topic Maps besonders zur Herstellung und Abbildung von Strukturen zwischen verschiedenen Ressourcen. Ansätze zur Integration von Daten aus beiden Modellen existieren[28].

[28] Vgl. http://www.zdnet.de/builder/program/0,39023551,39144931-1,00.htm

3.3.8. XOL

Ein weiteres, auf XML aufsetzendes Format zur Wissensrepräsentation, welches abschließend erwähnt werden soll, ist die XML-based Ontology exchange language (XOL), die 1999 entwickelt wurde. Im Gegensatz zu momentan verwendeten Ontologiebeschreibungssprachen setzt diese nicht auf RDF(S) auf, sondern direkt auf XML. Hintergrund ist, dass mithilfe von XOL keine Ontologien semantisch beschrieben werden sollen, sondern die Sprache in erster Linie dazu benutzt wird, Ontologien zwischen heterogenen Anwendungssystemen auszutauschen. Ein XOL-verfasstes XML-Dokument kann nach einem festen Schema genauso Informationen über Ontologieversion, sowie darin bezeichnete Klassen und Individuen enthalten, doch sind im Gegensatz zu den anderen in 3.3. vorgestellten Formaten keinerlei Inferenzoperationen auf diesen Informationen möglich. Bei der Entwicklung von OIL wurden dennoch wesentliche Aspekte aus XOL übernommen.

3.3.9. Layer-Architektur

Die vorausgegangenen Beschreibungen heutzutage verfügbarer Sprachstandards zur Bereitstellung von Meta-Informationen mit einer semantischen Beschreibung der Inhalte hat gezeigt, dass diese auf einer Kombination mehrerer Konzepte beruht. XML, RDF, RDFS und OWL kennzeichnen dabei verschiedene Abstraktionsebenen, die aufeinander aufbauen und sich gegenseitig ergänzen, wie in Abbildung 20 illustriert. Die Kombination dieser Techniken wird als wegweißend für die Repräsentation semantischer Informationen im Semantic Web angesehen und für nachfolgende Betrachtungen angenommen.

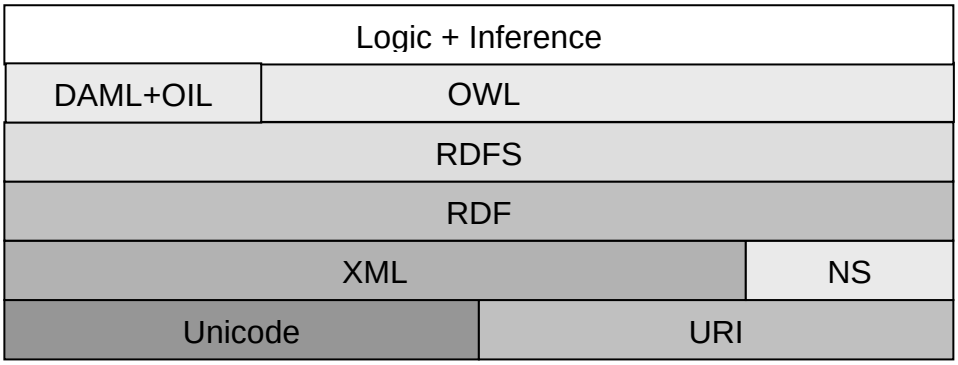


Abbildung 20: Semantisches Architekturmodell

3.4. Einbettung von Semantik in (X)HTML-Dokumente

3.4.1. Motivation

Mit XML, RDF, RDFS und OWL ist es prinzipiell möglich, die Bedeutung einzelner Konzepte maschinell erfassbar bereitzustellen. Die Idee des Semantic Webs geht davon aus, dass nach und nach immer mehr Webseiten mit semantischen Informationen versehen werden, und auf längere Sicht die Inhalte einzelner Seiten wie in einer großen verteilten Datenbank korrekt zueinander in Beziehung gesetzt werden können. Dies setzt voraus, dass die dafür benötigten Techniken so einfach und verständlich sind, dass sie ohne großen zusätzlichen Zeitaufwand auch von Benutzern ohne größere Programmiererfahrung eingesetzt und benutzt werden können – ob in einer expliziten Form oder irgendwo im Hintergrund verborgen, ohne dass der Benutzer das Wesen der semantischen Annotation erfassen muss. Betrachtet man beispielhaft die Benutzung einiger Ontologiebeschreibungen basierend auf RDF/XML, so lässt sich daran zweifeln, ob diese Technologien auf kürzere Sicht im Privatsektor zur Anwendung kommen könnten. Chen wirft in [Che07] diesbezüglich die Frage auf: „*Do you really believe companies can make money from products that require people to edit and query data like this…*“, indem er parallel dazu auf eine RDF-Datei mit einem Friend of a friend (FOAF)-Nutzerprofil verweist.

Ein weiteres Problem kommt sofort auf bei der Frage, wie Inhalte einer bestehenden (X)HTML-Seite mit RDF/XML-Tripeln ausgezeichnet werden sollen, sodass eine Maschine diesen das referenzierte Konzept zuordnen kann, ohne dass dies für den Nutzer ein Mehr an Aufwand bedeutet. Den Aufwand, den kompletten Inhalt einer Webseite redundant in einer RDF/XML-Datei zu halten, wird sicher niemand betreiben wollen.

3.4.2. SHOE

SHOE als Akronym für Simple HTML Ontology Extensions war ein erstes Projekt der University of Maryland[29] im Jahr 1996, welches als Erweiterung von HTML angedacht war, um Inhalte auf Webseiten mit semantischen Informationen auszeichnen zu können. Dazu wurde eine Auswahl zusätzlicher Tags vorgeschlagen, um damit Ontologien in Form von Konzeptbeziehungen und darauf definierten Eigenschaften abbilden zu können. Die Tags waren dabei proprietär und kein Bestandteil von HTML, konnten aber von SHOE-basierten Agenten gelesen und verarbeitet werden. Eine Webseite wurde dazu mit der Angabe *<META HTTP-EQUIV="SHOE" CONTENT="VERSION=1.0">* gekennzeichnet und darauf folgend die Inhalte mithilfe einiger Tags wie *<category></category>, <relation></relation>* oder *<instance></instance>* beschrieben, die sich auf eine SHOE-Ontologie bezogen, die unter dem Tag *<useontology/>* referenziert wurde. Seit 2002 gilt das SHOE-Projekt als ausgelaufen.

[29] Vgl. http://www.cs.umd.edu/projects/plus/SHOE/

3.4.3. Meta-Angaben

In der Spezifikation der Hypertext-Markup-Language ist explizit ein Tag zur Angabe beliebiger Metadaten-Angaben vorgesehen, welches seitdem zur unsichtbaren Einbettung verschiedenster allgemeiner aber auch unternehmensabhängiger Informationen in Webseiten verwendet wird. Offiziell beschrieben wird dieses mit folgendem Wortlaut:

„The META element is used within the HEAD element to embed document meta-information not defined by other HTML elements. Such information can be extracted by servers/clients for use in identifying, indexing and cataloging specialized document meta-information."

1996 erschien dazu vom W3C ein Vorschlag, wie Metadaten in HTML-Dokumente universell eingebettet werden können[30], sodass sie von jedem Internetbrowser auswertbar sind und die Einführung neuer, proprietärer Tags nicht erforderlich ist. Dazu wurde eine Konvention vorgeschlagen, im head-Bereich eines HTML-Dokumentes die <meta>-Auszeichnungsmarke zu benutzen, welche zwei Attribute *name* und *content* besitzen kann, wobei *name* die Bezeichnung für eine Metainformation darstelle und *content* den spezifischen Wert dieser Eigenschaft auf der betreffenden Seite. Pro Meta-Tag soll nur die Definition einer Meta-Information zulässig sein; mehrere Meta-Tags könnten aber aufeinander folgen. Dem Namen einer Metainformation sollte dabei eine Schemabezeichnung als Prefix, getrennt durch einen Punkt, vorausgehen (bspw. *DC.author*), wobei unter Benutzung dieses Schemabezeichners ein Verweis auf die Definition des Schemas angegeben werden konnte (*<link rel="SCHEMA.schema_identifier" href="url" />*). Neben der Bereitstellung allgemeiner zusätzlicher Informationen (beispielsweise Dublin Core) wird dieses System auch zur Inhaltserfassung durch andere Anwendungen (Platform Independent Content Selection (PICS)) verwendet.

3.4.4. Verlinkung auf externe RDF-Beschreibung

Auch wenn das redundante Halten von Inhaltsinformationen in einer HTML-Datei und einer externen RDF/XML-Datei wie in Abschnitt 3.4.1 bereits angesprochen zusätzliche Arbeit bereitet, so kann diese in manchen Anwendungsfällen gerechtfertigt sein. HTML bietet Möglichkeiten, eine externe Ressource maschinenlesbar zu referenzieren (*link*-Tag oder direkt als Hyperlink mit entsprechender *rel*- und *type* (*application/rdf+xml*)-Angabe).

[30] Vgl. http://www.w3.org/Search/9605-Indexing-Workshop/ReportOutcomes/S6Group2.html

3.4.5. Einbettung in Kommentarbereiche

Sofern die in den RDF statements beschriebenen Sachverhalte allgemeine Aussagen oder vorwiegend Aussagen über externe Ressourcen machen, wäre ein triviales Vorgehen, die RDF/XML-Repräsentation direkt und unverändert in ein (X)HTML-Dokument einzufügen. Dies ist zwar theoretisch möglich, woraufhin eine Validierung des entsprechenden Dokuments jedoch fehlschlagen wird. Praktisch werden RDF statements in diesem Fall häufig in einen HTML-Kommentarbereich eingeschlossen[31], wobei dies den gegenteiligen Effekt bringt, dass HTML-Kommentare in vielen Fällen bei der maschinellen Verarbeitung ignoriert werden. Das System lässt sich nicht nur auf RDF/XML-Beschreibungen anwenden, sondern wird beispielsweise auch zur Einbettung von entsprechenden Serialisierungen in Turtle (siehe Abschnitt 3.3.3) genutzt[32].

3.4.6. eRDF

Im Jahr 2006 gab es Bestrebungen, RDF-Tripel direkt in Form von embedded RDF[33] in Webseiten zu integrieren, indem sowohl die Möglichkeiten der Angabe von Dokumenteigenschaften in Meta-Tags als auch durch Auszeichnung einzelner Informationen im Dokumentbody in Kombination genutzt wurden. Im Gegensatz zu dem SHOE-Ansatz sollten dazu jedoch keine neuen Tags verwendet, sondern auf bestehende HTML-Elementattribute (*id, class, rel*) zurückgegriffen werden. Um als Dokument mit eingebetteten RDF-Informationen erkannt und behandelt zu werden, musste die Angabe *<head profile="http://purl.org/NET/erdf/profile">* im Seitenkopf vorhanden sein. Die Extraktion der RDF statements erfolgte in der Regel via XSLT mit einer in der Profildatei hinterlegten Transformationsvorschrift (profileTransformation)[34]. Die Verarbeitung der Dokumente lieferte brauchbare Ergebnisse, auch wenn nicht alle Aussagen, die in RDF formulierbar sind, damit abgebildet werden konnten. Inzwischen wurde eRDF durch neuere Konzepte abgelöst.

3.4.7. RDFa

Alle bisher vorgestellten Ansätze zur Einbettung von RDF statements in bestehende (X)HTML-Seiten boten interessante Ideen, hatten aber jedes für sich gewisse Nachteile. Unter der Zielstellung, eine RDF-Variante zu entwickeln, welche sich direkt in Dokumente einbetten lässt, die in einem spezifischen XML-Dialekt geschrieben sind, begann das W3C mit der Entwicklung von RDFa, welches in weiten Teilen die gleiche Zielstellung wie eRDF zur direkten Einbettung von Eigenschaftsbeziehungen in (X)HTML-Dokumente durch die Verwendung gültiger Tagattribute der XHTML-Spezifikation verfolgt.

---

[31] Eine Beispielanwendung ist die Einbettung von Copyrightinformationen durch die Creative Commons License, beschrieben unter http://www.ibm.com/developerworks/xml/library/x-think18.html, welche anschließend via GRDDL (http://www.w3.org/2003/12/rdf-in-xhtml-xslts/grokCC.xsl) extrahiert werden können

[32] Vgl. http://inamidst.com/sw/hturtle/

[33] Siehe http://research.talis.com/2005/erdf/wiki/Main/RdfInHtml

[34] Vgl. http://research.talis.com/2005/erdf/extract-rdf.xsl

Das aktuelle Referenzdokument dazu (working draft) stammt vom 26. Oktober 2007. RDFa verwendet sowohl Attribute aus der XHTML1.0 – Spezifikation (*href, rel, rev, content, datatype*) als auch einige wenige Attribute aus der zukünftigen XHTML2.0 – Spezifikation (*about, property, role*)[35], weswegen es nach Veröffentlichung einer XHTML2-Recommendation in Zukunft sicher stark an Bedeutung gewinnen wird. Ebenso ist wie bereits bei eRdf vorgesehen, Meta-Informationen über das Dokument selbst direkt als Meta-Tag-Angabe im *head*-Bereich der HTML-Seite spezifizieren zu können. Möglichkeiten existieren, womit RDFa-Beschreibungen aus Webdokumenten extrahiert werden können. Es handelt sich dabei in der Regel wieder um einen XSLT-Transformationschritt, andere Techniken sind aber ebenso denkbar, um RDF-Tripel aus bestehenden XML-Dokumenten abzuleiten (GRDDL - Gleaning Resource Descriptions from Dialects of Languages).

3.4.8. Microformats

RDFa stellt eine sinnvoll einsetzbare Möglichkeit dar, Objekte einer Webseite direkt mit semantischen Informationen annotieren zu können. Dazu werden nur wenige Attribute benötigt, deren Bedeutung dem Nutzer bekannt sein muss. Im engeren Sinne wird dadurch der Nutzer von der Anforderung befreit, überhaupt RDF statements schreiben zu müssen, sondern diese werden in einem Transformationsprozess (GRDDL) automatisch aus den vorhandenen Dokumentdaten extrahiert. Dennoch ist ein gewisses Grundverständnis über die internen Abläufe nötig (so muss der Benutzer beispielsweise das Wesen einer Ressource, die mit *about* referenziert wird, von dem einer Eigenschaft unterscheiden können). Eine seit 2005 laufende Entwicklung unter der Bezeichnung Microformats stellt darauf aufbauend die Frage, inwieweit diese Unterscheidung für einen Nutzer überhaupt von Belang sei. Eine semantische Auszeichnung von Internetseiten sei zwar wünschenswert, doch müsste dazu das Interesse des Nutzers im Mittelpunkt stehen, nicht das einer Maschine. Damit eine Technik weit verbreitet genutzt werde, müsste eine einfache Markierungsoperation ausreichen, um Anwendungen verständlich zu machen, um welches Konzept es sich bei einem konkreten Dokumentabschnitt handelt (*Datum, Name, Adresse, Telefonnummer, Kontonummer, …*), Zitat:

"Designed for humans first and machines second, microformats are a set of simple, open data formats built upon existing and widely adopted standards. Instead of throwing away what works today, microformats intend to solve simpler problems first by adapting to current behaviors and usage patterns" (http://microformats.org/about/)

[35] Bedeutung siehe http://www.w3.org/TR/xhtml2/mod-metaAttributes.html

Einfachheit steht bei Microformats im Mittelpunkt. Als Folge dessen wird ein Großteil der in 3.3.9. beschriebenen Layerarchitektur zur semantischen Beschreibung eines Konzepts aufgegeben und stattdessen altbekannte Methoden verwendet. Ähnlich wie bei eRdf werden dazu bestehende HTML-Tagattribute verwendet, im Speziellen das *class*-Attribut (häufig in Kombination mit dem *title*-Attribut). Darin vorkommenden Bezeichnern wird durch eine Spezifikation eine konkrete Semantik zugeordnet, die innerhalb einer Microformat-unterstützenden Anwendung implementiert wird. Statt wie bei RDFa auf eine dezentralisierte Verwaltung von Ontologien zu setzen („*jeder kann eigenständig Ontologien modellieren und den darin definierten Konzepten Eigenschaften zuordnen*“) werden Microformats zentral durch eine Entwickler-Community für bestimmte Problembereiche definiert[36]. Als Konsequenz daraus gibt es Anwendungsfälle, für welche keine passende Microformat-Beschreibung existiert (*80:20*-Regel). Microformats sind so generell wie möglich ausgelegt, um die Bedeutung immer wiederkehrender Informationen erkennen und durch Applikationen weiterverarbeiten zu können.

3.4.9. Vergleich von Microformats und RDFa

Microformats und RDFa gehen die gleiche Problemstellung aus zwei unterschiedlichen Richtungen an. Beide erlauben es, auf einfache Art und Weise die Inhalte einer Webseite mit Meta-Informationen über die konzeptionelle Bedeutung der Daten zu versehen, welche maschinenlesbar weiterverarbeitet werden können, gleichzeitig aber keine redundante Datenhaltung erfordern oder von dem menschlichen Benutzer eine längere Einarbeitungszeit abverlangen. Welche Technologie sich auf längere Sicht durchsetzen wird (ob überhaupt eine von diesen beiden, oder vielleicht eine ganz andere), ist bis zum heutigen Tag offen, allerdings zeigt eine nähere Betrachtung, dass Microformats und RDFa voneinander unabhängige Stärken besitzen, welche sich je nach Problemstellung gegenseitig ergänzen könnten.[37] Die nachfolgende Tabelle stellt beide Ansätze anhand unterschiedlicher Kriterien gegenüber, wobei ein Teil den Ausführungen von Evan Prodromou [Pro06] und Benjamin Nowack [Now07] entstammt.

Vom Wesen her ist dabei herauszustellen, dass zwei wesentliche Dinge den Einsatz von Microformats von RDFa unterscheiden. Zum Einen existiert bereits eine Reihe externer Anwendungswerkzeuge und Browsererweiterungen, welche Microformat-Annotationen von Webseiten extrahieren können, was für RDFa-Annotationen noch nicht der Fall ist, andererseits bietet nur RDFa die Möglichkeit, eigenständige Modellbeschreibungen für spezifische Anwendungsfälle zu verfassen und diese mit anderen bestehenden Ontologien zu kombinieren.

[36] Der Entwicklungsprozess eines Microformats ist unter http://microformats.org/wiki/process näher beschrieben

[37] Unter http://morethanseven.net/posts/what-erdf-can-learn-from-microformats/ wird Tom Morris zitiert, dass Microformats 80% aller Anwendungsfälle abdecken könnten, wo RDFa die restlichen 20% effektiv zu lösen vermag

	Microformats	RDFa
Zielsetzung		
Ziel	Einbettung semantischer Annotationen in bestehende HTML-Dokumente	Einbettung semantischer Annotationen in bestehende XML-Dokumente
Zielgruppe	Mensch	Maschine
Anwendungsdomain	Eingeschränkt	Universell
Wissensdarstellbarkeit	Eingeschränkt	Mächtig
Skalierbarkeit	Schlecht	Gut
Fokus	Einfachheit	Universalität
Motivation	Kommerziell	Wissenschaftlich
Entwicklung		
Zentralisierung	Zentraler Ansatz, Entwicklungsprozess in einer Community, bis Microformat einen Status als Standard erhält	Dezentraler Ansatz, jeder kann eigene Beschreibungsmodelle entwickeln oder existierende Modelle für seinen Anwendungsfall nutzen
Entwicklung	Communitygetrieben	Forschungsgetrieben
Verbreitung	Beginnende Benutzung im Mainstream-Bereich	Bisher nur im Forschungsbereich eingesetzt
Unterstützung	Vielfach durch Anwendungen (z.B. Dreamweaver) und Erweiterungen (Mozilla) unterstützt	ausgereifte Werkzeuge zur XML-Verarbeitung, noch keine Applikationen mit explizier RDFa-Unterstützung existent
Spezifikation		
Spezifikation	Inoffiziell, ad-hoc („Tag soup“)	RDFa Primer als aktueller working draft, Aussicht auf Standardisierung
Dokumentation	Alle Microformats zentral gut dokumentiert, Mappings vorhanden	Spezifikation vorhanden, RDFa – Profil bei anderer Institution hinterlegt
Support	Große Community	Kein Support
Kombination mit anderen Beschreibungen	Unsicher, mehrere Microformats in dem gleichen Dokument zu verwenden	Problemlose Verwendung mehrerer Ontologien in einem Dokument
Stabile Syntaxspezifikation	Weitestgehend	Ja
Implementierung		
Auszeichnungsmittel	Auswahl an existierenden HTML-Attributen (class, title, rel)	Speziell zur Auszeichnung von Meta-Informationen vorhandene Tags der XHTML1.0 (href, rel, rev, content, datatype)- und XHTML2.0 (about, property, role)-Spezifikation
Einbettung in HTML	durch Benutzung der entsprechenden Klassenbezeichnungen, Angabe eines XMDP-HTML Profils empfohlen	RDFa Profile und Namespace-Angaben nötig, anschließend Benutzung als Property-Eigenschaften
Namespace	Keine explizite Benutzung von Namespaces vorgesehen	Beliebige XML Namespaces
Compliance	Abwärtskompatibel zu allen bisherigen HTML-Versionen, XHTML1.0 und auch das zukünftige XHTML2.0	Validierbar nur bez. XHTML2.0
Bedeutungsanalyse	Bedeutung durch Standarddokumente festgelegt, Beschreibung verwendeter Attributwerte in XMDP möglich, Umsetzung in Anwendungsprogrammen codiert	Bedeutungen und Beziehungsgefüge in Ontologiebeschreibung maschinenlesbar hinterlegt
Informationsgewinnung	Basierend auf Text-Analyse / DOM-Parsing, Umwandlung via XSLT in RDF-Statements (GRDDL) eingeschränkt möglich	GRDDL (XSLT)
Verweis auf andere Ressourcen / Aussagen über externe Ressourcen	Nur in passendem Kontext (vcard-picture, XFN-rel)	Jederzeit, grundlegendes Prinzip
Domainübergreifende Datenauswertung	Nein	Ja

Tabelle 5: Vergleich von Microformats und RDFa

So wäre es im vorliegenden Fall, eine semantische Beschreibung für Projekte, Prozesse und Aktivitäten zu entwerfen sinnvoll, diese Objekte und damit verbundene Eigenschaften voneinander unterscheiden zu können, beispielsweise ob es sich bei der gelesenen Information um das Startdatum eines Projekts handelt oder um einen Termin, zu dem eine konkrete Aktivität innerhalb dieses Projekts durchgeführt wurde. Microformats bieten dazu keine Möglichkeit, können beide Angaben jedoch so auszeichnen, dass sie in einem Kalenderformat mit einem kurzen Beschreibungstext direkt in andere Anwendungen exportiert werden können. Um beide Angaben maschinell unter einem getrennten Blickwinkel maschinell weiterverarbeiten zu lassen, eignet sich RDFa besser, weswegen prinzipiell eine Kombination beider Ansätze möglich erscheint und diesem (zumindest momentan) nichts entgegen spricht.

Zu erwähnen ist dabei abschließend, dass inzwischen Untersuchungen durchgeführt wurden, wie eine Verbindung zwischen Microformats und RDFa hergestellt werden kann. Ein Ergebnis davon ist hGRDDL [38], wodurch mit Microformats annotierte Webseiten in Webseiten mit RDFa-Inhalt transformiert werden können, wobei dies momentan wieder via XSLT realisiert wird. Statt direkt zu versuchen, aus Microformats eine RDF-Beschreibung via GRDDL abzuleiten, wäre dies eine Möglichkeit, über einen Zwischenschritt bessere Transformationsergebnisse geliefert zu bekommen[39].

3.5. Nutzbare Ontologien

Im Folgenden soll die Verfügbarkeit bereits verfügbarer Anwendungsdomain-Beschreibungen untersucht werden, die als Microformat oder in RDFa benutzbar sind. Kapitel 2 untersuchte dazu grundlegende Konzepte, welche in einem semantikbasierten Projekt- und Workflowmanagementsystem abbildbar sein müssen. Dazu zählen:

1. Kontakt- und Personeninformationen
2. Projektbeschreibungen
3. Prozessbeschreibungen
4. Aktivitätsinformationen
5. Dokumentbeschreibungen

Hinzu kommen Anforderungen, die Mitarbeiter (stakeholder) innerhalb eines Projektes referenzieren zu können sowie Abläufe und Zusammenhänge in Projekten und Prozessen (Zustandsübergänge und Abhängigkeiten) darstellen zu können. Es existiert eine Reihe weiterer Informationen, welche unter dieser Problemstellung eine Rolle spielen, beispielsweise Angaben dazu, mit welchen Hilfsmitteln eine konkrete Aktion auszuführen ist oder ausgeführt wurde (Task&Tool-Ontologien).

[38] Vgl. http://www.w3.org/2006/07/SWD/wiki/hGRDDL_Example

[39] bisher benötigt jedes Microformat ein eigenes Profil, in dem eine spezifische Transformationsdatei zur Umwandlung des verwendeten Microformats in RDF/XML hinterlegt ist, z.B. http://www.w3.org/2002/12/cal/glean-hcal

Die letztgenannte Anforderung sei im Folgenden aufgrund des sehr speziellen Charakters jedoch von der weiteren Betrachtung ausgeblendet.

a) Microformats

Die wohl am weitesten verbreitete Anwendung für Microformats stellen die Formate hCard und hCalendar dar, welche als offener Standard vorliegen. hCard basiert dabei auf der Beschreibung von VersitCards (vCards) , einer Art Visitenkartenformat zur Angabe persönlicher Kontaktinformationen (RFC2426), hCalendar auf dem iCalendar-Standard (RFC2445) zur Angabe von Terminen und Ereignissen an einem bestimmten Datum. Daneben existieren einige Microformat-Standardisierungen, welche auf dem *rel*-Attribut eines HTML-Elements basieren. Mit der XHTML Friends Network (XFN) Spezifikation können Beziehungen zwischen Personen abgebildet werden, die über einen Homepageverweis identifizierbar sind. Rel-License ermöglicht die Auszeichnung von Verweisen als Copyright-Information, Rel-Tag im Kontext des Web2.0 die Auszeichnung von Tags, welche den Inhalt der Webseite charakterisieren und mit dem Rel-Nofollow Microformat sollen Informationen hinterlegt werden können, welche einen Verweis als irrelevant für die Gewichtung des aktuellen Seiteninhalts auszeichnen.

Ebenso existieren eine Reihe an Vorschlägen (Drafts) für weitere Microformats, beispielsweise hAtom zur Auszeichnung fortlaufender Inhalte, hResume zur Auszeichnung von Lebensläufen, hReview zur Beschreibung von Produktzusammenfassungen oder xFolk zur Kennzeichnung von Bookmark-Sammlungen. hDoap ist ein Microformat, welches zur Beschreibung von Softwareprojekten verwendet werden und DOAP (Description of a Project)-Vokabular eingebettet in Webseiten abbilden kann. Die Entwicklung des letztgenannten Formats wurde jedoch auf private Initiative vorangetrieben[40].

Ansonsten befinden sich eine ganze Reihe weiterer Vorschläge zu Microformats in einem Entwicklungsprozess, der unter microformats.org/wiki gut unter dem Abschnitt Exploratory Discussions dokumentiert ist. Die Entwicklung orientiert sich an der Suche nach möglichst allgemeingültigen Anwendungsbeispielen aus der Praxis, die die Bereitstellung eines eigenen Microformats dafür sinnvoll erscheinen lassen (vgl. 80:20-Regel). Ein Brainstorming-Prozess kommt hinzu, welche Ausdruckskraft ein neu zu entwickelndes Microformat besitzen muss und auf welchen bereits existierenden Standards dies aufsetzen könnte. Die Seite http://microformats.org/wiki/exploratory-discussions zeigt, dass es in zurückliegender Zeit bereits viele Vorschläge gab, deren Entwicklung inzwischen aber zum Großteil wieder eingestellt oder abgebrochen wurde.

[40] Vgl. http://usefulinc.com/doap/

So wäre zur Abbildung von Projektmanagement ein Task Microformat interessant, von dem es zwar bereits einen konzeptionellen Vorschlag gibt[41], der auf vToDo als Teil der iCalendar-Spezifikation aufbaut, und damit für viele Anwendungsfälle ausreichend bereits in hCalendar abgebildet ist. Ähnlich verhält es sich mit Microformats zur Auszeichnung speziellerer Projekt- und Dokumenteninformationen. Diskussionen über die sinnvolle Nutzbarkeit derartiger Erweiterungen finden statt[42], laufen in vielen Fällen aber letztendlich auf das Ergebnis hinaus, dafür bereits bestehende Microformats zu verwenden.

Bei der Entwicklung eines semantikbasierten Projektmanagementsystems wird diesem Vorschlag gefolgt, und zur Auszeichnung von Informationen von allgemeinem Interesse für externe Anwendungen („Semantic Clipboard“) die gängigen Microformats hCard und hCalendar genutzt, während die Programmlogik der Anwendung basierend auf einer applikationsspezifischen Ontologie entworfen wird, welche mithilfe von RDFa in die einzelnen Webseiten eingebettet ist.

b) RDFa

In RDFa sind beliebige semantische Informationen in eine Webseite einbettbar. Dabei kann auf eine ganze Reihe bereits existierender RDF Schemas und anderweitig formulierter Ontologien zurückgegriffen werden, welche sich in der Vergangenheit bewährt haben und so eine Kompatibilität mit anderen Anwendungen sichergestellt wird, ohne dass die verwendeten Ontologien zentral administriert und deren Ausdruckskraft ohne Erweiterungsmöglichkeit vorgegeben wird.

Um sich einen ersten Überblick über im Netz verfügbarerer Ontologien zu verschaffen, eignen sich die Webseiten www.schemaweb.info und swoogle.umbc.edu recht gut. Es finden sich eine Reihe weiterer Kataloge über vorhandene Ontologiebeschreibungen im Netz (bspw. http://protege.stanford.edu/download/ontologies.html oder http://www.daml.org/ontologies/), in denen sich eine Suche jedoch als sehr aufwändig gestalten kann und deren Inhalte mitunter nicht mehr aktuell gehalten sind oder neuere Standards umfassen.

So finden sich hier ohne langen Suchaufwand Verweise auf ein RDF Schema zur Abbildung von vCards sowie Verweise auf die Verwendung von Dublin Core zur Beschreibung von Dokumenten, aber auch andere bekannte Beschreibungen wie RDF Calendar. Schwieriger gestaltet sich die Suche nach einer bereits existierenden Domain Ontology zur Beschreibung von Projekt- und Workflowmanagement, welche drei Anforderungen erfüllt:

[41] hToDo, siehe http://microformats.org/wiki/htodo

[42] Interessant diesbezüglich ist unter Anderem die Diskussion über ein Microformat hProject unter http://microformats.org/discuss/mail/microformats-discuss/2006-June/004596.html

1. sie ist allgemein genug, zur Implementierung eines Projektmanagementsystems verwenden werden zu können indem sie sich nicht nur auf die ökonomische Sicht beschränkt
2. sie ist speziell genug, um einfach nutzbar zu sein (Usability-Reusability trade-off)
3. sie erfuhr bereits eine gewisse Verbreitung, sodass eine Interoperabilität mit anderen Systemen in Zukunft sichergestellt ist.

Obwohl in verschiedenen Ontologie-Verzeichnissen eine ganze Reihe von Projektmodellen gefunden werden konnten, erfüllte keine der Ontologien alle drei Bedingungen. Viele Entwicklungen stellten sich als zu speziell heraus, welche im Rahmen kleinerer Universitätsprojekte entworfen wurden oder nur als Application Ontologies für eine spezifische Anwendung konzipiert wurden. Ontologien, welche die Kombination von Projekten mit Prozessen ermöglichten, wie dies aus einer objektorientierten Sicht möglich sein sollte, fanden sich keine. Als vielversprechendste Entwicklung stellte sich die Business Management Ontology (BMO) der Firma Jenz&Partner heraus[43], welche als Protegé-Beschreibung in OWL vorliegt. Da diese zur Umsetzung eines einfach gehaltenen Projektmanagementsystems jedoch bereits zu komplex und generell erschien, wurde im weiteren der Versuch unternommen, eigenständig eine leichtgewichtige Ontologiebeschreibung zur Abbildung von Projekt- und Prozesseigenschaften zu entwickeln.

3.6. Entwurf einer geeigneten Ontologie

Die Entwicklung neuer Ontologien in den Vorstellungen des Semantic Web ist nichts Außergewöhnliches. So schildert James Hendler in einer Veröffentlichung zu „*Agents and the Semantic Web*“, dass nach seinen Vorstellungen das Semantic Web nicht aus einer Auswahl an fertigen, durch Institutionen festgelegten und zentral verwalteten Ontologien bestehe, sondern vielmehr durch einen dezentralen Ansatz geprägt sein soll, in dem Nutzer wie heutzutage bereits auch jederzeit eigene Inhalte und in Zukunft auch eigene Ontologien veröffentlichen können. Die Folge ist eine große Anzahl kleinerer Ontologien, welche gegenseitig aufeinander verweisen können und stetig weiterentwickelt werden [Hen01]. Um eigenständig eine Ontologie für eine bestimmte Problemstellung zu entwerfen, gibt es ähnlich wie in der objekt-orientierten Programmierung verschiedene Ansätze, von denen sich Top-Down-zentrierte Methoden ebenso wieder finden wie Vorschläge basierend auf einem Bottom-Up-Ansatz ausgehend von bereits existierenden Ontologien. In [Bre07, Kap.8ff] werden dazu einige Methoden zur Ontologieentwicklung zusammengetragen. So wird in einem als Skelettmethode bezeichnetem Ansatz vorgeschlagen, ähnlich wie im Software Engineering durch Use Cases praktiziert, zunächst den Einsatzzweck einer Ontologie zu identifizieren und darauf zunächst als textuelle Beschreibung Konzepte und Beziehungen zwischen diesen Konzepten zu definieren (Top Down), welche anschließend formal dargestellt werden können.

[43] http://www.bpiresearch.com/Resources/RE_OSSOnt/re_ossont.htm

Ein ähnliches Vorgehen schlagen Noy und McGuiness basierend auf ihren Erfahrungen bei der Entwicklung von Protegé und Ontolingua vor, legen den Fokus jedoch stärker in den Bereich der Wiederverwendung bestehender Ontologien (Bottom Up). Ihr Modell des „*Ontology developments 101*“ umfasst folgende Schritte, die sich in erster Linie an Nutzer richten, welche nicht im akademischen Bereich tätig sind:

1. Bestimme Anwendungsbereich und –einfluss der zu entwickelnden Ontologie
2. Ziehe in Betracht, bereits bestehende Ontologiebeschreibungen wieder zu benutzen.
3. Zähle wesentliche Begriffe auf, welche in der neuen Ontologie abgebildet werden sollen.
4. Definiere davon ausgehend Klassen und eine Klassenhierarchie.

Nachfolgend seien diese Anforderungen beschrieben:

Die Anwendungsdomain der zu entwickelnden Webapplikation liegt im Bereich des Projektmanagements. Im Gegensatz zu klassischen Projektmanagementsystemen können Projekte weiter strukturiert sein und aus weiteren Teilprojekten, aber auch Prozessen bestehen, wodurch auch Methoden des Workflowmanagements in dem System eine Rolle spielen. Prozesse können ebenso in weitere Prozesse strukturiert sein oder aus elementaren Aufgaben, so genannten Aktivitäten, bestehen. Jedes dieser Elemente besitzt einen Titel, eine Statusinformation sowie die Möglichkeit, einen kurzen Beschreibungstext anzugeben. Einem Projekt können darüber hinaus Mitarbeiter (stakeholder) zugeordnet werden, welche für bestimmte Aufgaben innerhalb dieses Projektes verantwortlich sind. Dazu werden Projekten, Prozessen und Aktivitäten bestimmte Berechtigungen für diese Nutzer oder Nutzergruppen zugeordnet. Projekte, Prozesse und Aktivitäten besitzen weitere konzeptspezifische Eigenschaften: Ein Projekt hat sowohl ein geplantes Start- und Enddatum als auch die Möglichkeit, davon abweichende reale Start- und Endtermine einzutragen. Ebenso kann eine Information angegeben werden, inwieweit die Projektinformationen öffentlich anonymen Nutzern bereit gestellt werden können. Ein Prozess besitzt nur ein Start- und Enddatum der Ausführung, eine Aktivität hingegen wiederum einen geplanten Termin und einen tatsächlichen Durchführungstermin. Wie aus dem Workflowmanagement bekannt, können zwischen einzelnen Elementen Abhängigkeiten in Form von Kanten definiert werden, wobei dazu Informationen gespeichert werden müssen, wie die Prozessabarbeitung bei mehreren parallel ein- (join) oder aus- (split) gehenden Kanten voranschreiten soll. Ebenso ist es sinnvoll, in der Entwurfsumgebung der Anwendung Darstellungsinformationen über die Position einzelner Objekte speichern zu können, ähnlich wie dies in einigen Workflowspezifikationen wie XPDL bereits realisiert wird.

Basierend auf den Ausführungen stellen Projekte, Prozesse und Aktivitäten das zentrale Konzept innerhalb der benötigten Ontologie dar. Da diese auf semantischer Ebene zwar verschiedene Konzepte abbilden, aus Entwicklersicht prinzipiell aber gleichermaßen verwaltet werden können, wäre eine Ableitung von einer Oberklasse sinnvoll (ein Projekt kann aus weiteren Teilprojekten oder Prozess bestehen, die wiederum Prozessbeschreibungen oder Aktivitäten enthalten können). Der Oberklasse, hier exemplarisch mit *SemProjObject* bezeichnet, könnten anschließend Eigenschaften zugeordnet werden, welche alle drei abgeleiteten Konzepte gleichermaßen enthalten (Titel, Beschreibung, Status, Erstellungsinformationen, …). Ebenso könnten Abhängigkeiten zwischen einem Projekt und einem Prozess oder einer Aktivität sehr einfach abgebildet werden, wenn alle möglichen Konzepte der gleichen Oberklasse angehören. Unter Hinzunahme weiterer Konzepte könnte eine Ontologie entwickelt werden, wie sie in Abbildung 21 etwas vereinfacht dargestellt ist.

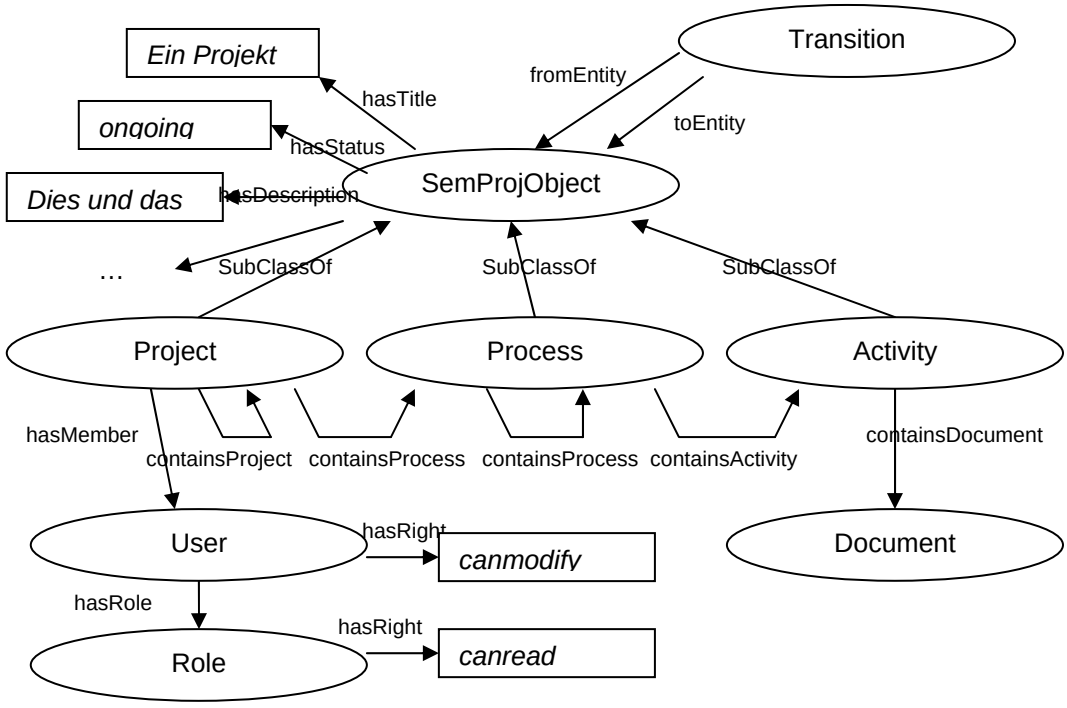


Abbildung 21: Naives Modell einer Projekt- und Workflowmanagementontologie

Unabhängig von der Fragestellung, wie aussagekräftig das dargestellte Modell ist (jedes Modell kann nur einen Teil der Realwelt ausschnittsweise abbilden), bringt diese Modellierung zwei Probleme mit sich, welche bei dieser Herangehensweise nicht sofort offensichtlich sind. Zum einen existieren an verschiedenen Stellen des Modells Eigenschaften mit der gleichen Bezeichnung (*semproj:containsProcess; semproj:hasRight*). Dies mag aus menschlicher Erfahrung heraus problemlos möglich zu sein, bereitet jedoch sofort Schwierigkeiten, wenn es in einem RDF Schema abgebildet werden soll, da dadurch die Forderung nach einer eindeutigen, widerspruchsfreien maschinellen Abbildung verletzt ist.

So ist einer Eigenschaft in RDFS nur eine Konzepteinschränkung (*rdfs:domain*) zuordbar, nicht aber die Einschränkung auf mehrere Konzepte, die nicht aus einer gemeinsamen Oberklasse abgeleitet werden können. Für einen Nutzer ist dies ohne Hintergrundwissen nicht unbedingt nachvollziehbar, und er würde bei Problemen wahrscheinlich direkt die Eigenschaftsbezeichnungen ändern, sodass diese dem Namen nach unterschieden werden können. Das zweite Problem ist längerfristig problematischer, da durch die Übertragung der Anfordernisse aus Implementiersicht wesentliche Eigenschaften der Konzepte Projekt, Prozess und Aktivität miteinander in einem Oberkonzept vermischt werden. Besser wäre es, diese möglichst unabhängig voneinander beschreiben zu können, sodass ein Projektstatus beispielsweise durch eine Anwendung problemlos von einem Aktivitätsstatus unterschieden werden könnte. Sollten im aktuellen Entwurf weitere Anforderungen und Eigenschaften hinzukommen, wäre möglicherweise eine systemweite Änderung nötig. Dies ist dadurch bedingt, dass hier im Grunde genommen eine Applikations-spezifische Ontologie entworfen wurde und keine Domain-spezifische Ontologie. Wird für jedes der Konzepte eine eigene Ontologie zur Beschreibung benutzt, die vielleicht bereits von einer anderen Person im Netz entwickelt und bereitgestellt wurde, kann dieses Problem umgangen werden. Gleichzeitig wird durch die Verwendung verschiedener Namespaces das erstangesprochene Problem gelöst, da sich die Eigenschaft *project:containsProcess* nun direkt von der Eigenschaft *process:containsProcess* unterscheiden würde. Geht man davon aus, dass das System eigenständige Ontologie zur Beschreibung von Projekten, Prozessen, Aktivitäten, Nutzern, Dokumenten und anderen Dingen verwendet, so könnten diese in einem zweiten Schritt mit einer leichtgewichtigen Abhängigkeitsbeschreibung zueinander in Beziehung gesetzt werden. Diese nur zu Systemzwecken benutzte applikationsnahe Ontologie kann weitere Vereinfachungen bezüglich der Eigenschaften ermöglichen, indem Referenzen zu Teilprojekte, Teilprozessen oder enthaltenen Aktivitäten nun direkt als *semproj:containsEntity* beschrieben werden, ohne die konzeptionelle Trennung aufgeben zu müssen. Diese Vorgehensweise wird im Folgenden zur Implementierung angewendet und sei abschließend noch einmal zusammengefasst:

- Es existieren fünf zu modellierende Konzepte im Rahmen des zu entwickelnden Systems: Nutzer, Projekte, Prozesse, Aktivitäten und Dokumente
- Nutzerinformationen werden mithilfe einer bereits existierenden Ontologie zur Auszeichnung von vCards verwaltet
    - Namespace: http://www.w3.org/2001/vcard-rdf/3.0#
    - RDF Schema: http://www.w3.org/2001/vcard-rdf/3.0
- Zur Auszeichnung von Dokumentinformationen werden die bereits weit verbreitete Dublin Core – Beschreibungsattribute benutzt
    - Namespace: http://purl.org/dc/elements/1.1/
    - RDF Schema: http://purl.org/dc/elements/1.1/

- Es werden (sehr einfach gehaltene) RDF Schemas zur Beschreibung von Projekten, Prozessen und Aktivitäten bereitgestellt
- Die Projekt-Ontologie enthält die Eigenschaften *title, plannedstartdate, plannedenddate, actualstartdate, actualenddate, description, ispublic, status, createdon, createdby, lastmodifiedon, lastmodifiedby*, sowie eine Auflistung an Nutzern, welche an dem Projekt beteiligt sind mit entsprechenden Rollen- und Rechtezuordnungen
    - Namespace: http://vsr.informatik.tu-chemnitz.de/projects/2007/semproj/project#
    - RDF Schema: http://vsr.informatik.tu-chemnitz.de/projects/2007/semproj/project.rdfs
- Eine Prozess-Ontologie enthält die Eigenschaften *title, startdate, enddate, description, status, createdon, createdby, lastmodifiedon, lastmodifiedby*, sowie Aussagen über die Rechte der am übergeordneten Projekt beteiligten Nutzer
    - Namespace: http://vsr.informatik.tu-chemnitz.de/projects/2007/semproj/process#
    - RDF Schema: http://vsr.informatik.tu-chemnitz.de/projects/2007/semproj/process.rdfs
- Die Activity-Ontologie enthält die Eigenschaften *title, planneddate, actualdate, description, status, createdon, createdby, lastmodifiedon, lastmodifiedby*, sowie Aussagen über die Rechte der am übergeordneten Projekt beteiligten Nutzer
    - Namespace: http://vsr.informatik.tu-chemnitz.de/projects/2007/semproj/activity#
    - RDF Schema: http://vsr.informatik.tu-chemnitz.de/projects/2007/semproj/activity.rdfs
- Als Bindeglied existiert eine weitere anwendungsspezifische Ontologie mit Informationen, welche nur zu Darstellungs- und Informationszwecken innerhalb der Anwendung relevant sind. Dazu zählen Hierarchieinformationen, Positionsangaben der Elemente (*semproj:positiontop, semproj:positionleft*), Split/Join-Informationen und Abhängigkeitsangaben.
    - Namespace: http://vsr.informatik.tu-chemnitz.de/projects/2007/semproj#
    - RDF Schema: http://vsr.informatik.tu-chemnitz.de/projects/2007/semproj.rdfs

Die kompletten RDF Schemas sind im Anhang der Arbeit zu finden.

4. Implementierung eines Prototypen

“Never send a human to do a machine's job” – The Matrix, Agent Smith, 1999

4.1. Festlegung der Systemumgebung

Während wohl die Mehrzahl bereits entwickelter, semantikorientierter Webapplikationen auf einer Umsetzung in Java unter Zuhilfenahme des Jena frameworks[44] beruht, soll der Prototyp des zu entwickelnden Projektmanagementsystems auf einer Implementierung als ASP.NET 2.0 – Webanwendung basieren.

Die Entwicklung erfolgt daher unter dem Microsoft Visual Studio 2005. Neben dem installierten .NET 2.0 framework werden weiterhin die ASP.NET Ajax Erweiterungen (System.Web.Extensions.dll) sowie das ASP.NET Ajax Control Toolkit (AjaxControlToolkit.dll) benötigt. Zur Umsetzung darüber hinausgehender, in einer Rich Internet Application sinnvoll einzusetzende Funktionen wurde außerdem die Benutzung des ASP.NET Ajax Future Releases in Erwägung gezogen (Microsoft.Web.Preview.dl), in welchem beispielsweise zusätzliche Möglichkeiten zur clientseitigen Behandlung von Drag&Drop-Effekten bereitstehen, dies aufgrund festgestellter Inkompatibilitäten mit bestehenden Controls jedoch wieder verworfen.

Als Ausführungsumgebung wird eine Windows 2003 – Distribution mit einem Internet Information Server (IIS) 6.0 oder höher empfohlen. Besondere Hardwareanforderungen werden nicht gestellt, es wird jedoch von einer praxistauglichen Serverkonfiguration (Prozessor neuerer Generation mit mind. 1.5GHz, Arbeitsspeicher von mindestens 512MB, Internetanbindung mit mind. 2MBit/s) ausgegangen. Auf Clientseite sollte die Webapplikation in einem aktuellen Internet-Browser (Internet Explorer 6.0 oder höher, Mozilla und alle anderen auf der Gecko-Engine basierenden Browser ab Version 1.0, Opera ab Version 7.6. u.a) auf jedem handelsüblichen PC lauffähig sein. Bedingt durch den Einsatz von AJAX (Asynchronous Javascript and XML) muss dazu die Unterstützung von Javascript aktiviert sein. Ebenso muss die Annahme von Cookies erlaubt sein, da einige Daten in einer Nutzersession hinterlegt sind.

Die Installation eines speziellen Datenbankserver wie MS SQL 2005 zur Verwaltung der Daten wird nicht benötigt. Der Prototyp der Anwendung wird so konzipiert, dass alle Daten zusammen mit semantischen Metainformationen in dem gleichen Dokument hinterlegt werden. Dies bedeutet, jedes einzelne, durch das Projektmanagementsystem verwaltete (X)HTML-Dokument stellt einen eigenständigen Datenspeicher dar, in dem alle benötigten Informationen vorhanden sind.

[44] Beziehbar über http://jena.sourceforge.net/

Hintergrund ist der, dass in Zukunft vorstellbare Suchmaschinenagenten beliebiger Institutionen das „Semantic Web“ nach passenden Informationen durchsuchen könnten und diese dabei zur Lokalisierung der Daten nacheinander verschiedene Internetressourcen anhand ihrer URL aufrufen. Was als Ergebnis zurückgeliefert wird, ist eine normale HTML-Datei, ob nun dynamisch generiert oder statisch auf dem entsprechenden Webserver vorhanden, deren Inhalt daraufhin ausgelesen und weiterverarbeitet werden kann. Im Rahmen dieser Arbeit soll dazu untersucht werden, wie aufwändig sich die Datenhaltung innerhalb einzelner Dateien gestalten kann und wie performant der Verarbeitungsprozess durch externe Applikationen vonstatten geht. Inwieweit dieses Vorgehen sinnvoll ist, soll abschließend im Ergebnisteil der Arbeit diskutiert werden.

Die genaue Spezifikation der technischen Anforderungen findet sich in dem Dokument „Analyse und Spezifikation des Systems“ im Anhang der Diplomarbeit.

4.2. Semantische Frameworks

Zur Verarbeitung von als RDF vorliegenden Informationen empfiehlt sich der Einsatz spezialisierter RDF frameworks, wobei für verschiedene Programmiersprachen unterschiedliche Implementierungen existieren. Während sich beispielsweise in Java eine ganze Reihe an frameworks findet, welche das Auslesen und die Verarbeitung von RDF statements unterstützen, ist die Auswahl an .NET frameworks deutlich geringer. Drei frameworks wurden dabei in einer Voruntersuchung in die engere Auswahl bezogen:

- Die Redland RDF API und Raptor RDF Parser Library von Dave Beckett (http://librdf.org/)
- Die OwlDotNetApi von Bram Pellens (http://users.skynet.be/bpellens/OwlDotNetApi/index.html)
- Die SemWeb-library von Joshua Tauberer (http://razor.occams.info/code/semweb/)

Die Redland RDF API schien bei einem Vergleich der drei frameworks zunächst den größten Funktionsumfang zu bieten, da diese neben der RDF-Generierung und Abfrage (SPARQL/RDQL) auch weitere relevante Funktionen wie einen expliziten Support für GRDDL bereitstellte und für verschiedenste Plattformen bereits seit 2000 entwickelt wird. Die Benutzung innerhalb des Visual Studios gestaltete sich jedoch als schwierig, da die .NET-Unterstützung nur innerhalb der Mono Entwicklungs- und Laufzeitumgebung bereitgestellt wurde und bei einem Einsatz im Visual Studio nicht alle entsprechenden Dateien kompatibel waren oder portiert werden konnten.

Die OwlDotNetApi basierte auf dem Projekt DriveRDF und erweiterte dies um Möglichkeiten, explizit Owl-Ontologien unter .NET2.0 schreiben und verwalten zu können. DriveRDF wurde in vielen Quellen im Voraus als leistungsfähiges Framework erwähnt, dessen Entwicklung unter www.driverdf.org aber bereits Mitte 2007 eingestellt zu sein schien.

OwlDotNetApi war dem gegenüber zwar noch verfügbar und die Nutzung innerhalb des Visual Studios gestaltete sich problemlos, allerdings stellte sich heraus, dass sich das framework lediglich auf das programmgesteuerte Auslesen von RDF statements und das Schreiben von OWL-Dateien beschränkte und darüber hinaus keine weiteren Möglichkeiten zur Verarbeitung bot.

Letztendlich wurde sich für den Einsatz der SemWeb library (Version 1.01) entschieden, welche alle benötigten Funktionalitäten und darüber hinaus eine übersichtliche und gute Dokumentation bietet, gleichzeitig aber auch problemlos unter C#/.NET2.0 genutzt werden kann. Bereit gestellt wird unter anderem ein einfaches Werkzeug, ein einfaches Reasoning basierend auf RDFS auf mehreren RDF statements durchführen zu können. Die explizite Unterstützung von OWL-beschriebenen Ontologien wird leider nicht geboten, doch wurde bereits in der Vorbetrachtung des zu entwickelnden Projektmanagementsystems deutlich, dass deren Komplexität und Ausdruckskraft nicht unbedingt benötigt wird (weswegen nur i.W. nur RDFS-Beschreibungen betrachtet wurden).

Die mittels RDFa eingebetteten RDF statements wurden mithilfe einer XSL Transformation (GRDDL) aus den einzelnen XHTML-Dokumenten extrahiert. Zur Transformation wurde dazu ein (momentan noch in Entwicklung befindliches) Profil des „*Instituts National de Recheché en Informatique et en Automatique*“ (Ingria) unter http://ns.inria.fr/grddl/rdfa/ verwendet.

Dieser Ansatz bietet wesentliche Flexibilität, da zum einen die RDF statements direkt aus den (X)HTML-Dateien extrahiert werden können und nicht in separaten RDF stores physisch hinterlegt werden müssen, andererseits der Transformationsprozess uniform abläuft, unabhängig davon, unter Zuhilfenahme welches Modells die Daten innerhalb des entsprechenden Dokuments ausgezeichnet wurden. Bisher mussten dazu verschiedene Profile verwendet werden, die Informationen enthielten, wie beispielsweise Dublin Core- Beschreibungen, vCard-Informationen oder Microformat-Annotationen in eine Menge uniformer RDF statements transformiert werden können. Entsprechende Angaben zu dazu benötigten Mappings finden sich unter http://esw.w3.org/topic/CustomRdfDialects oder http://esw.w3.org/topic/CustomRdfDialects/GrddableMicroformats.

4.3. Wünschenswerte Funktionalitäten

Vor der Implementierung des Prototypen wurden in einer Analysephase mehrere Anforderungen an den Leistungsumfang des zu entwickelnden Systems gestellt, welche nachfolgend zusammengefasst werden sollen:

- Auf die Projektverwaltung des Systems sollen nur berechtigte Nutzer zugreifen können, welche sich durch Angabe eines Benutzernamens und Passworts bei einem Identitätsprovider authentifizieren
- Mit dem System sollen Projekte auf verschiedenen Abstraktionsebenen verwaltet werden können. Projekte können dabei als ein Container aufgefasst werden, welcher aus weiteren Teilprojekten, darin enthaltenen Prozessen und weiteren elementar durchzuführenden Aufgaben besteht. Jedes der Elemente soll eigenständig anhand einer URL referenzierbar sein.
- Auf die Datenbasis kann über diese URL jederzeit anonym zugegriffen, die darin enthaltenen Daten aber nicht verändert werden (Agenten externer Applikationen können die Projektdaten weiterverarbeiten). Falls erforderlich, kann der Zugriff jedoch eingeschränkt werden,
- Das Projektmanagementsystem stellt eine grafische Editorumgebung zur Verfügung, über welche neue Projekte modelliert und bestehende Projekte verändert werden können. Dabei wird auf bereits existierende Projektdaten zurückgegriffen, indem die semantischen Informationen aus den betreffenden Dokumenten ausgelesen und durch das System direkt verarbeitet werden. Die einzelnen Objekte in dem Editor können neu erstellt, dynamisch per Drag&Drop verschoben und gelöscht werden. Ebenso können Verbindungen (Kanten) zwischen mehreren Objekten eingezeichnet werden
- In dem Editor existiert ein Eigenschaftsfenster, über welches die Eigenschaften jedes Objekts verändert werden können. Titel und Status eines Projektes, Prozesses oder einer Aktivität sind direkt änderbar.
- Jedes Objekt besitzt ein Symbol, worüber der Nutzer in eine niedrigere Abstraktionsebene wechseln kann, welche daraufhin geladen und dargestellt wird
- Bei der Erstellung eines neuen Objektes besteht die Möglichkeit, die Daten aus bereits existierenden Ressourcen zu übernehmen. Bodendorf [Bod03, p.131] identifiziert dazu verschiedene Vorgehensweisen, um Inhalte in einem Dokumentenmanagementsystem im Falle eines „Case Reuse“ zu übernehmen, wobei diese im Wesentlichen in ein einfaches *Copy*-Verfahren und einen *Adapt*ionsansatz unterschieden werden können. Es wird sich in diesem Fall für eine Kopie der bestehenden Ressourcen und Zuordnung zu dem neuen Projekt entschieden

- Durchzuführenden Aktivitäten können weitere Dokumente zugeordnet werden, beispielsweise textuelle Beschreibungen, hochladbare Dokumente oder Verweise auf externe Dokumente (Linklisten). Diese Dokumente können als begleitende Hinweise zur durchzuführenden Aufgabe als auch als Ergebnis nach Fertigstellung der jeweiligen Aufgabe angesehen werden.
- Es ist ein einfaches Benachrichtigungssystem zu implementieren, wann Aufgaben von einer zuständigen Person fertig gestellt wurden und der Arbeitsprozess voranschreiten kann. Die Benachrichtigung erfolgt in Form einer Email[45], welche durch das System verschickt wird. Die dazu benötigten Kontaktdaten kann der entsprechende Nutzer auf einer eigenen Kontaktseite hinterlegen, aus deren RDFa-Annotation sich das System die entsprechenden Informationen herleitet.
- Alle inhaltlichen Änderungen von Projekten, Prozessen, Aktivitäten, Dokumenten und Nutzerinformationen sollen inkrementell gespeichert werden und darauf ein rudimentäres Versionsmanagementsystem abgebildet werden, mit dem frühere Versionen angesehen und gegebenenfalls wieder hergestellt werden können.
- Die institutionsübergreifende Dokumentenverarbeitung, bei denen jeweils eine eigene Instanz des Projektmanagementsystems betrieben wird, wird prinzipiell ermöglicht

Eine komplette Spezifikation der Soll- und Kann-Anforderungskriterien des Systems findet sich im Anhang „Analyse und Spezifikation des Systems“.

4.4. Systemarchitektur

Im Folgenden seien wesentliche Aspekte der Systemarchitektur ausgehend von den Anforderungen aus Kapitel 4.3. näher beschrieben. Eine schematische Darstellung der Systemarchitektur findet sich ebenso im Anhang der Diplomarbeit.

- Verzeichnisarchitektur

Abbildung 22 illustriert schematisch die dem Projektmanagementsystem zugrunde liegende Verzeichnishierarchie. So finden sich im Wurzelverzeichnis der Anwendung die Ordner

[45] Unter http://www.jdk.de/de/ecmblog/tag/workflow wird eine alternative Möglichkeit diskutiert, Informationen über eingetretene Ereignisse einem Nutzer besser über einen RSS feed bereitzustellen, da automatische Benachrichtigungen per Email nach kurzer Zeit nicht mehr ausreichend beachtet und als störend empfunden werden können. Eine Extraktion und Bereitstellung der Daten des Systems als RSS feed ist prinzipiell möglich.

- *access* (enthält Webseiten, welche nur nach entsprechender Authentifikation des Nutzers aufgerufen werden können),
- *bin* (enthält alle notwendigen dll-Dateien),
- *data* (enthält die gesamte Datenbasis des Projektmanagementsystems),
- *layout* (für das Frontendlayout benötigte Grafiken),
- *modules* (enthält alle verwendeten Controls und damit verbundenen Anwendungscode)
- *Web References*
- *Webservice* (asmx-Dateien des SemProj-Webservices sowie der AJAX-Webmethods)

sowie alle öffentlich aufrufbaren Webseiten (*default.aspx*) als auch weitere anwendungsspezifische Dateien (Master-Seiten-Definition, *Web.config, Web.sitemap*).

Dem Ordner *data* kommt dahingehend besondere Bedeutung zu, da darin alle vom System verwalteten Daten hinterlegt sind. Projektseiten, Prozessinformationen, Aktivitätsangaben als auch die Kontaktseiten einzelner Nutzer und Seiten mit Dokumentinhalten werden in jeweils separaten Verzeichnissen Form einzelner HTML-Dateien verwaltet. Diese existieren je nach Anzahl der bisherigen Änderungen mehrmals getrennt nach letztmaligem Änderungsdatum, um die Anforderung des Versionsmanagements auf einfache Weise umsetzen zu können.

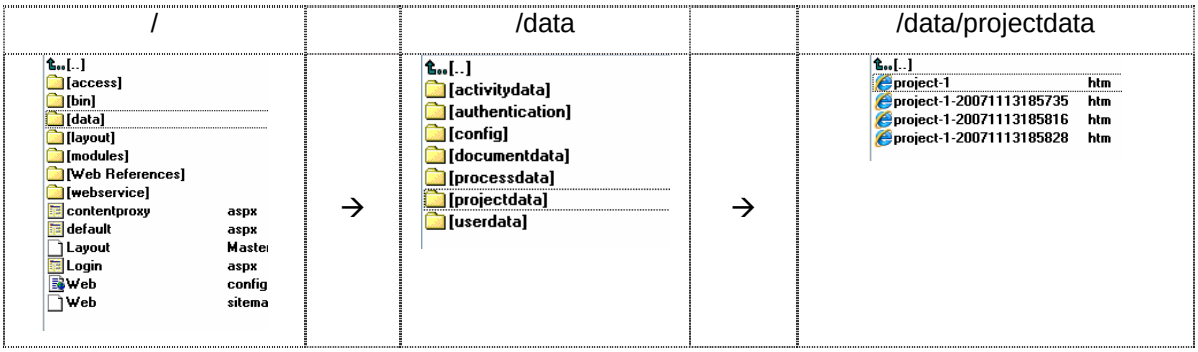


Abbildung 22: Verzeichnishierarchie des Projektmanagementsystems

Wie bereits angesprochen, enthält jede der HTML-Dateien sowohl eigentliche Daten über das jeweilige dargestellte Konzept (Titel, Datumsangaben, Beschreibungstexte, bei Projekten/Prozessen Illustration der weiteren Hierarchie in Form einzelner Symbole) als auch semantische Annotationen in Form von RDFa und Microformats, welche durch das System extrahiert und zur Bearbeitung weiterverwendet werden können, wie Abbildung 23 schematisch zeigt.

Diplomarbeit André Langer
Typ: project
Titel: Diplomarbeit André Langer
Status: ongoing
Geplantes Startdatum: 01.05.2007
Geplantes Enddatum: 31.12.2007
Öffentliches Projekt: yes
Beschreibung: Diplomarbeit zum Thema "Semantic Web - basiertes System zur Unterstützung von Workflow- und Projektmanagement"
Erstellt am: 23.11.2007
Erstellt von: langa
Zuletzt geändert am: 23.11.2007
Zuletzt geändert von: langa
Mitarbeiter: langa (Diplomand)
<?xml version="1.0" ?>
Diplomarbeit André Langer
Diplomarbeit zum Thema "Semantic Web - basiertes System zur Workflow- und Projektmanagement"
23.11.2007
Diplomand

Abbildung 23: Ausschnitt aus HTML-Datei und daraus abgeleiteten RDF statements

- Authentifikation + Nutzerrollen

Benutzer werden anhand einer gültigen Benutzername - Passwort – Kombination authentifiziert, welche durch einen Identitätsprovider überprüft wird. Dies kann im einfachsten Fall ein Webservice sein, der beide Daten in verschlüsselter Form entgegennimmt, und einen booleschen Wert zurückliefert, ob die eingegebenen Daten mit den Informationen aus einer hinterlegten Datenbasis übereinstimmen oder nicht. Das Verfahren ist praktikabel, bietet jedoch einige Schwächen und kann durch bessere Methoden ersetzt werden. Eine davon stellt das Identity Federation System (idFS) dar, welches dazu benutzt werden kann, Webapplikationen und Web Services mit Security tokens basierend auf WS-Federation und WS-Security zu sichern [Gae05]. Die zukünftige Integration dieses Systems zur Nutzeridentifizierung in die zu entwickelnde Anwendung wird vorgesehen.

Die erfolgreiche Authentifizierung wird Cookie-basiert gespeichert und der Zugriff auf den eingeschränkten Bereich der Anwendung erlaubt. Im zweiten Schritt der Authentifikation wird einem Nutzer bei jedem Anmeldevorgang eine gleiche, eindeutige ID zugeordnet, über welche das System Informationen über unterschiedliche Nutzer unterscheiden und verwalten kann. Basierend auf dieser ID kann der Nutzer eine eigene Kontaktseite anlegen, welche über eine URL (z.B. *http://www.example.or/data/userdata/user-1.htm*) aufrufbar ist und über welche der Nutzer gleichzeitig eindeutig referenziert werden kann. Weiterhin erhält der Nutzer nicht nur eine System-ID, sondern ebenso eine Systemrolle zugeordnet. Das Projektmanagementsystem unterscheidet zwischen drei verschiedenen Rollen:

- *Standardbenutzer*: darf an bestehenden Projekten teilnehmen und diese modifizieren
- *Projektleiter*: darf neue Projekte und neue Nutzer anlegen
- *Administrator*: darf systemweite Änderungen vornehmen sowie die Gesamtheit an Nutzern und Projekten administrativ verwalten.

Alle System-Informationen bezüglich des angemeldeten Nutzers werden innerhalb einer Session zum schnelleren Zugriff gespeichert. Berechtigungen für neue Nutzeraccounts können von einem Nutzer in der Rolle des Projektmanagers oder Administrators jederzeit hinzugefügt, geändert oder entzogen werden. Ein System, bei dem sich neue Nutzer eigenständig anmelden und daraufhin die Anwendung ohne autoritäre Freigabe direkt nutzen können, ist zum jetzigen Zeitpunkt nicht vorgesehen.

- Webservice

Um ein kooperatives Arbeiten zwischen parallel betriebenen Instanzen des Projektmanagementsystems ermöglichen zu können, werden alle dateisystemnahen Operationen über einen Webservice gekapselt, der mittels WS-Security vor unbefugtem Zugriff gesichert ist. Die Bereitstellung verschiedener Zugriffsmethoden in einem Webservice wurde weiterhin erforderlich, da durch den Einsatz von AJAX design patters mehrere Operationen von Clientseite aus angestoßen werden, welche anschließend in einem asynchronen Kommunikationsprozess mit dem Server im Hintergrund laufen und durch einen Webservice bereitgestellte Methoden aufrufen. Diese bereitgestellten Funktionen umfassen im Speziellen Möglichkeiten, eine Auflistung aller auf dem Server verwalteter Projekt-URLs und Nutzer zurückgeliefert zu bekommen, als auch Änderungen an Projekten, Prozessen, Aktivitäten, Dokumenten oder eigenen Nutzerdaten speichern zu können. Administrative Aufgaben können dem gegenüber aus Sicherheitsgründen nur lokal ausgeführt werden.

- Ausführungs-Engine

Um nicht nur die Erstellung und Modellierung von Projektstrukturen in einer grafischen Editorumgebung bereitzustellen, sondern auch ein Projektmanagement im engeren Sinne anzubieten, sind Funktionen erforderlich, um auf den vorhandenen Projektdaten bestimmte Operationen ausführen zu können. Im Workflowmanagementbereich sind Implementierungen dazu bereits gut erforscht und in einem Workflow-Referenzmodell der Workflow Management Coalition (WfMC) näher spezifiziert. Kern dieses Models ist die Trennung eines Workflowmanagementsystems in einzelne Module zur Modellierung, Steuerung, Ausführung und Überwachung von Prozessen, wobei mehrere Schnittstellen definiert werden, bei denen eine austauschbare Workflowbeschreibungssprache (im engeren Sinn XPDL) und mehrere Workflow Engines im Mittelpunkt stehen. Bei einfachen Projektmanagementsystemen ist diese modulare Trennung nicht zu finden, bietet sich bei dieser Entwicklung in einfacher Weise jedoch an, da sowohl eine maschinell auswertbare Projekt- und Prozessbeschreibung vorhanden ist, als auch eine Strukturierung der einzelnen Projekte und Prozesse durch optionale Kanten, welche Zustandsübergänge repräsentieren.

Teil der Systemarchitektur ist dazu eine Komponente, welche (ähnlich wie eine Workflow Engine) den aktuellen Ausführungszustand innerhalb eines Projektes feststellen und für einen spezifischen Nutzer eine Aussage treffen kann, welche Tätigkeiten er als nächstes auszuführen hat und inwieweit diese Ausführung von vorausgehenden Aktivitäten abhängt, deren Ausführung der Nutzer selbst nicht beeinflussen kann. In diesem Fall soll der Nutzer den Status einer Aktivität, welche durch eine andere Person oder Personengruppe auszuführen ist, natürlich nicht ändern können.

Zu diesem Zweck wurde eine (sehr einfach gehaltene) Ausführungs-Komponente implementiert. Diese benutzt gerade die in einem Projekt, Prozess oder einer Aktivität nun per RDFa hinterlegten semantischen Informationen über Zeitraum und Status sowie daran beteiligten Mitarbeitern und damit zueinander in Beziehung stehenden übergeordneten Projekten oder Prozessen. Dies wird dahingehend erschwert, dass innerhalb einer Hierarchieebene Abhängigkeiten zwischen Projekten, Prozessen und Aktivitäten zugelassen wurden und die Ausführung einer Tätigkeit von mehreren vorausgehenden Tätigkeiten abhängig sein kann. Will van der Aalst identifizierte dazu an die 40 Muster (Workflow patterns), welche durch ein Workflowmanagementsystem unterstützt werden müssten, um jeden beliebigen Workflow damit abbilden zu können[46]. Die Ausführungsengine des vorliegenden Projektmanagementsystems beschränkt sich dabei lediglich auf einfache Split- und Join-Operationen, welche eine Aussage darüber machen, ob die folgende Aktion erst ausgeführt werden kann, wenn alle vorausgehenden Aktionen ausgeführt worden (*AND*-Join) oder die Ausführung einer vorherigen Tätigkeit ausreicht (*OR*-Join, Split analog). In der Implementierung des Systems wurde weiterhin zugelassen, dass es auf einer Hierarchieebene mehrere disjunkte Projekt- und Prozessverkettungen geben kann. In diesem Fall werden alle Teilgraphen eigenständig auf ihre Ausführbarkeit hin überprüft.

- GRDDL – Transformationsfunktionen

Innerhalb der entwickelten Anwendung wurden weitere Funktionen bereitgestellt, welche aus Projektmanagementsicht uninteressant sind, aber den Einsatz und die korrekte Funktionsweise bei der Extraktion semantischer Informationen zu Testzwecken zeigen. Der Transformationsprozess besteht dabei aus zwei Stufen. In der ersten Stufe wird eine normale XSL Transformation (GRDDL) zur Gewinnung einzelner RDF statements durchgeführt, deren Ergebnis in einem zweiten Schritt mithilfe des verwendeten SemWeb frameworks aufbereitet und zusammengefasst wird. Beide Transformationsschritte sind über eine Proxy-Funktion im Wurzelverzeichnis der Anwendung überprüfbar. Am unteren Bildrand des Projekteditors existiert weiterhin eine Schaltfläche zur direkten Anzeige der momentan verarbeiteten RDF Tripel.

[46] Vgl. http://www.workflowpatterns.com/

4.5. Lizenzmodell

Das entwickelte Projektmanagementsystem wird mit der Bezeichnung "*SemProj*" unter einer Berkeley Software Distribution (BSD) – Lizenz zur zukünftigen Weiterentwicklung bereitgestellt. Software unter einer BSD-Lizenz darf frei verwendet, kopiert und weiter vertrieben werden. Ebenso ist eine Modifikation der bestehenden Codebasis und Weiterentwicklung möglich. Dies schließt eine Verwendung des Quellcodes als Vorlage für die Entwicklung kommerzieller Software nicht aus. Der weiterentwickelte Quellcode muss dabei nicht zwingenderweise öffentlich zur freien Verfügung bereitgestellt werden, wie dies beim zugrunde liegenden Quellcode im Rahmen der BSD-Lizenz getan wurde. Lediglich der Copyright-Vermerk innerhalb des ursprünglichen Projekt-Quellcodes darf nicht verändert oder sogar entfernt werden. Seit 1999 muss im Rahmen der Weiterverbreitung der Name des ursprünglichen Entwicklers nicht mehr zwingend benannt werden (3-clause BSD).[47]

[47] Vgl. http://www.opensource.org/licenses/bsd-license.php

5. Praxistest

„Work expands (so as) to fill the time available for its completion" ~ The Economist, C. N. Parkinson, 1955

5.1. Installation

Die Installation des semantikbasierten Projektmanagementsystems stellt keine besonderen Anforderungen und entspricht der normalen Vorgehensweise der Veröffentlichung (deployment) eines ASP.NET 2.0-Projekts auf einem Internet Information Server unter Windows 2003 oder höher. Lediglich die ASP.NET AJAX Extensions assembly muss im Global Assembly Cache (GAC) installiert sein. Alle weiteren zur Ausführung benötigten dlls sind dem Quellcode im Anhang beigefügt. Das Verzeichnis */data* sowie alle darunter liegenden Verzeichnisse müssen mit Schreibberechtigungen für den ASP.NET-Nutzer ausgestattet sein. Ebenso muss ein Lesezugriff auf die darin enthaltenen Daten basierend auf einem Web-Request sichergestellt sein.

Als Erweiterung zu der SemProj-Webapplikation wird wie in Absatz 4.4. beschrieben, das Identity Federation System benutzt, welches im Unterordner idFS hinterlegt ist und innerhalb de Internet Information Servers als eigenständige Anwendung registriert werden muss.

Zum Abgabetermin der Diplomarbeit existiert eine lauffähige, fertig eingerichtete Testumgebung der Anwendung unter

http://www.andré-langer.de.

5.2. Performance

Bei einem ersten praktischen Einsatz des Systems stellte sich überraschenderweise die anfängliche Performance als sehr problematisch heraus, weswegen an einzelnen Stellen Modifikationen vorgenommen werden mussten. Es kam dabei zu längeren Verarbeitungszeiten bei der Extraktion der RDF – Tripel aus einem semantisch annotierten (X)HTML-Dokument, wobei selbst bei strukturell einfachen Dokumenten der GRDDL-Transformationsprozess bei bis zu gemessenen 2,0 Sekunden lag. Wie in Kapitel 4.4 beschrieben schloss sich daran ein weiterer Transformationsschritt zur Aggregation und Aufbereitung der RDF statements durch das verwendete RDF framework an, bevor die Daten durch einen Reasoner verarbeitet werden konnten. Dieser Optimierungsschritt fiel aus Performancesicht jedoch nicht weiter ins Gewicht.

Hauptproblem dabei war, dass besonders auf der Projektübersichtsseite eines Nutzers eine Vielzahl an Webressourcen verarbeitet werden müssen, um darauf nutzerrelevante Informationen zu extrahieren, weswegen die Webanwendung zu Beginn praktisch nur sehr schwerfällig und aufgrund der langen Wartezeiten kaum nutzbar war.

Um diese Problematik zu beherrschen, wurden zwei Maßnahmen durchgeführt. Zum einen wurden alle Inferenzoperationen programmatisch kontrolliert durchgeführt und auf eine lokale Datenbasis beschränkt. Nachdem die Zugehörigkeit eines Mitarbeiters zu einem konkreten Projekt bestimmt wurde, wurden nur noch darin direkt referenzierte Dokumente auf ihre Relevantheit hin durch die Ausführungs-Engine untersucht, wodurch das System insgesamt besser skalierte. Weiterhin wurden alle RDFa→RDF Transformationsergebnisse in Abhängigkeit der referenzierten Ressource in einem Cache dynamisch zwischengehalten, solange die Ressourceinformationen durch einen Änderungsvorgang nicht modifiziert wurden. Damit konnte auf bereits extrahierte RDF Tripel aus einem Dokument bei Wiederverwendung wesentlich schneller zugegriffen werden, ohne die RDF statements in einer separaten Datei getrennt von dem eigentlichen Webdokument physisch speichern zu müssen.

Die Konsequenz daraus ist, dass besonders zu Beginn das entwickelte Projektmanagementsystem eine spürbar höhere Verarbeitungszeit benötigt, um alle Daten initial zu erfassen, als dies im weiteren Verlauf der Fall ist. Das System ist für einfache Projektmanagementanwendungen dennoch gut benutzbar, sollte allerdings insgesamt als Prototyp betrachtet werden, der für längerfristige Anwendungen in der Praxis unter dieser Zielstellung neu implementiert werden sollte, da dieses Problem zu Beginn der Entwicklung nicht in der Gesamtheit absehbar war.

5.3. Bedienung

Die Bedienung des Projektmanagementsystems orientiert sich an einer Aufteilung, wie sie während der Evaluation in der Praxis bewährter Webapplikationen als üblich und funktionell geeignet festgestellt werden konnte. Die Navigationsstruktur ist flach und in einem Zustandsdiagramm in der Spezifikation im Anhang ausführlich dargestellt. Ausgangspunkt bei Aufruf der Startseite des *SemProj* – Projektmanagementsystems ist eine öffentliche Seite, auf der knapp das System beschrieben ist und momentan laufende, als öffentlich einsehbar gekennzeichnete Projekte aufgelistet werden. Links im Inhaltsbereich findet sich ein Navigationsmenu, in dem der Benutzer zunächst nur den Menupunkt „*Anmeldung*“ auswählen kann, da die weitere Benutzung des Projektmanagementsystems nur autorisierten Benutzern gestattet ist. Der Link verweist auf eine weitere Seite, auf der in einem Anmeldeformular *Benutzername* und *Passwort* eingegeben werden können. Die eingegebenen Nutzerdaten werden daraufhin durch den SemProj-IdentityProvider des Identity Federation Systems (idFS) überprüft. Nur dem System bekannte Benutzer können sich hierbei anmelden. Eine Registrierung neuer Nutzer ist nicht vorgesehen; diese können nur durch einen Projektmanager oder Administrator im Backend hinzugefügt werden.

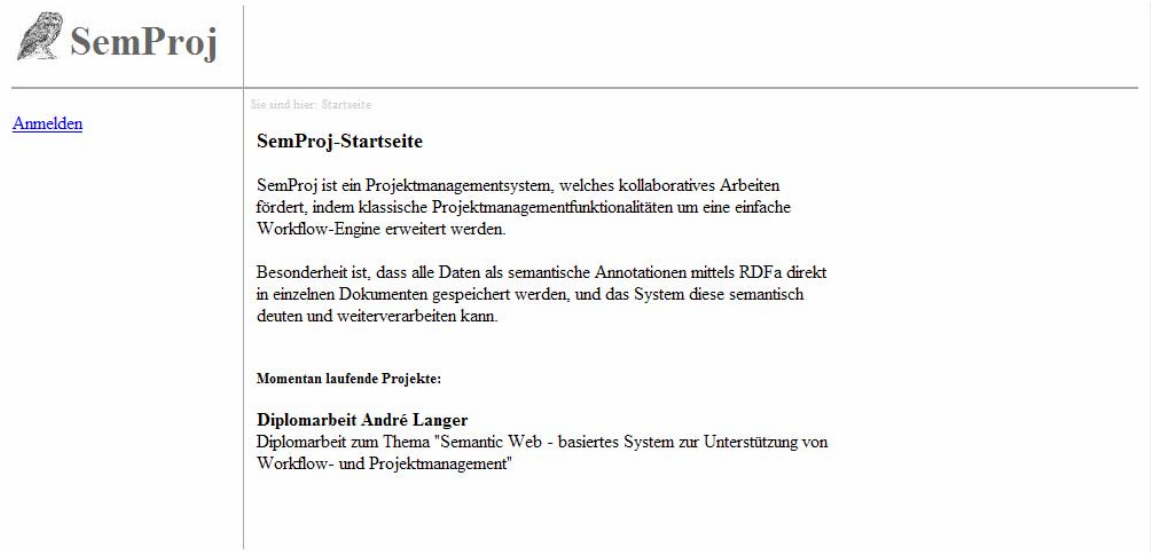

SemProj
Anmelden
Sie sind hier: Startseite
SemProj-Startseite
SemProj ist ein Projektmanagementsystem, welches kollaboratives Arbeiten fördert, indem klassische Projektmanagementfunktionalitäten um eine einfache Workflow-Engine erweitert werden.
Besonderheit ist, dass alle Daten als semantische Annotationen mittels RDFa direkt in einzelnen Dokumenten gespeichert werden, und das System diese semantisch deuten und weiterverarbeiten kann.
Momentan laufende Projekte:
Diplomarbeit André Langer
Diplomarbeit zum Thema "Semantic Web - basiertes System zur Unterstützung von Workflow- und Projektmanagement"

Abbildung 24: Startseite von SemProj

Nach erfolgter Anmeldung erhält der Benutzer im Navigationsmenu die Möglichkeit, eine eigene Nutzerseite mit Kontaktdaten anzulegen. Dort können der eigene Name, Berufsbezeichnung, ein Foto, Emailadressen, Büroanschrift, Telefonnummer und Internetadresse hinterlegt werden. Die Informationen werden mit semantischen Annotationen versehen und werden vom Projektmanagementsystem zur Kontaktaufnahme benutzt, können aber auch durch andere Nutzer mithilfe geeigneter Browsererweiterungen (bspw. *Tails* oder *Operator* unter Mozilla) exportiert werden.

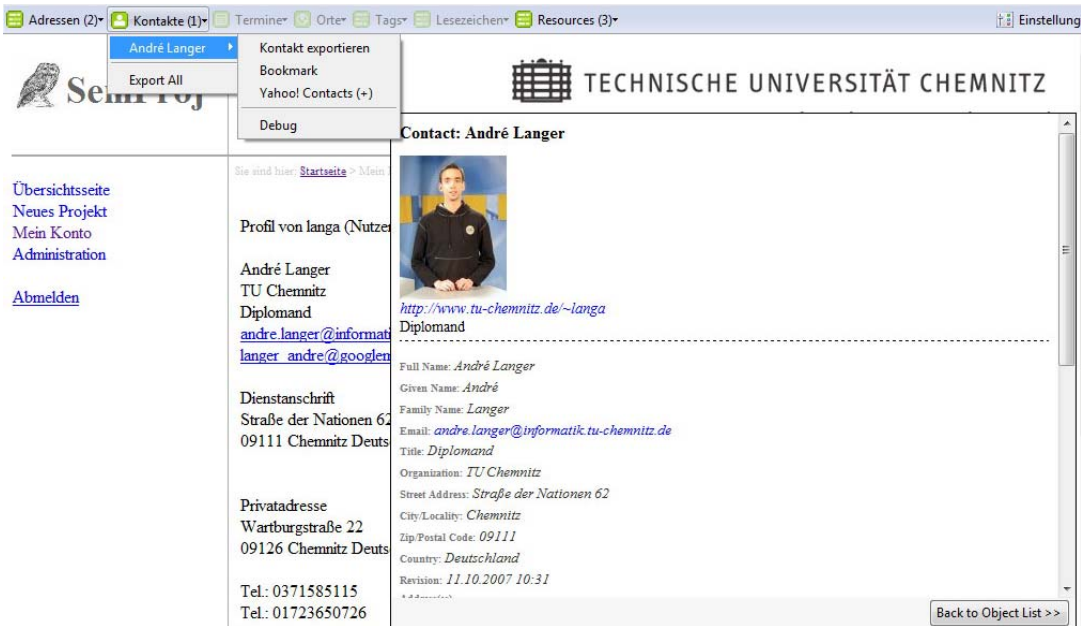


Abbildung 25: Kontaktseite eines Nutzers mit Exportmöglichkeit

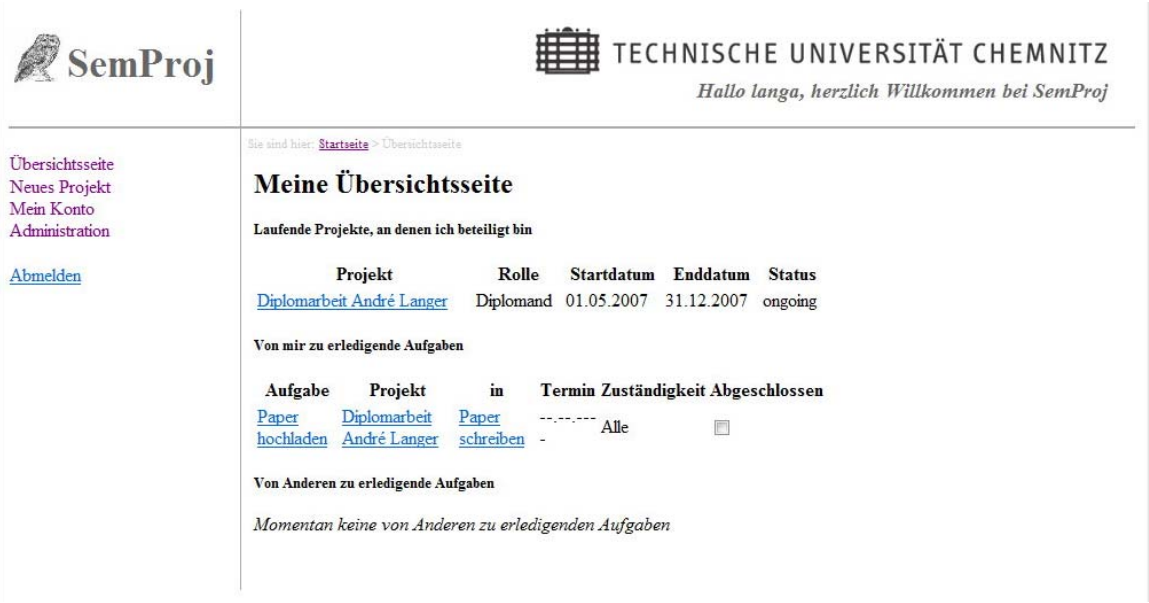

SemProj
TECHNISCHE UNIVERSITÄT CHEMNITZ
Hallo langa, herzlich Willkommen bei SemProj
Übersichtsseite
Neues Projekt
Mein Konto
Administration
Abmelden
Sie sind hier: Startseite > Übersichtsseite
Meine Übersichtsseite
Laufende Projekte, an denen ich beteiligt bin
Projekt Rolle Startdatum Enddatum Status
Diplomarbeit André Langer Diplomand 01.05.2007 31.12.2007 ongoing
Von mir zu erledigende Aufgaben
Aufgabe Projekt in Termin Zuständigkeit Abgeschlossen
Paper hochladen Diplomarbeit André Langer Paper schreiben --.--.---- - Alle
Von Anderen zu erledigende Aufgaben
Momentan keine von Anderen zu erledigenden Aufgaben

Abbildung 26: Zentrale Übersichtsseite

Besonderer Bedeutung kommt im Weiteren einer zentralen Übersichtsseite zu, auf welcher alle laufenden Projekte und alle als nächstes zu erledigenden Aufgaben aufgelistet sind. Aufgaben können über eine Checkbox direkt als erledigt markiert werden, wodurch sich der Status der entsprechenden Aktivität auf „*durchgeführt*“ (*completed*) verändert. Ansonsten ist die Übersichtsseite ein Einstiegspunkt zu allen damit verbundenen Projekten und Prozessen, welche durch Klick auf einen Link ausgewählt werden können und auf eine entsprechende Repräsentation im Projektmanagementeditor verweisen.

Der Projektmanagementeditor ist zweigeteilt. Während sich in der oberen Hälfte eine Auflistung aller Eigenschaften des momentan betrachteten Projektes, Prozesses, Aktivität oder Dokuments findet, wird darunter die weitere Workflowstruktur dargestellt. Unterobjekte sind durch einfache Ikonen repräsentiert, welche einer vereinfachten Form der Business Process Modelling Notation (BPMN) nachempfunden wurden und somit leicht verständlich sein sollten. Da diese jedoch nur die Modellierung von Prozessen vorsieht, wurde für die Darstellung von Projektentitäten eine analoge Repräsentation gewählt, wobei alle Objekte zusätzlich beschriftet sind. Auf die Implementierung von Pools und Swimlanes zur Darstellung getrennter Zuständigkeiten wurde aus Platzgründen verzichtet. Stattdessen sind Objekte, die der Nutzer selbst manipulieren kann, als aktiv gekennzeichnet, während er für Aktionen anderer Nutzer keine Einstellungen vornehmen kann. Welcher Nutzer für die Ausführung einer Aktion zuständig ist, kann auf einer Detailseite nachgesehen und ggf. verändert werden. Dahinter steht ein einfaches Rechtesystem, wobei für jedes Objekt festgelegt werden kann, ob ein Systembenutzer die Details eines Projekts, Prozesses oder einer Aktivität lesen darf, Einstellungen für dieses Objekt anpassen darf oder für die Erfüllung der entsprechend dadurch repräsentierten Aktion zuständig ist.

Die Umsetzung der Rechte wird durch eine in 4.4. beschriebene Ausführungs-Engine realisiert, wobei Objekte, auf welche der Nutzer keinen Zugriff hat, ausgegraut werden. Ebenso werden Objekte dunkel dargestellt, welche in Bezug auf den aktuellen Projekt-/Workflowstatus noch nicht bearbeitbar sind, weil vorausgehende Prozesse erst abgeschlossen werden müssen. Alle anderen, aktiven Aufgaben werden mit einer Hintergrundfarbe unterlegt, welche den aktuellen Status widerspiegelt, ob Termingrenzen eingehalten werden (*grün*), die Fertigstellung nötig ist (*orange*) oder geplante Termine überschritten sind (*rot*). Zusätzlich findet sich am unteren Rand eines Ikons eine Fortschrittsanzeige, welche prozentual die Anzahl der erfüllten Aufgaben im Verhältnis zu den insgesamt zu erfüllenden Aufgaben darstellt. Um den Status eines Projektes, eines Prozesses oder einer Aktivität zu ändern, findet sich weiterhin eine Auswahlliste, in welcher der zuständige Benutzer direkt eine Aufgabe als *„begonnen“, „abgeschlossen“ oder „abgebrochen“* kennzeichnen kann.

Um ein neues Projekt anzulegen, kann ein Benutzer mit der Berechtigung eines Projektmanagers oder Administrators im Navigationsmenu den Punkt „*Neues Projekt*“ auswählen. Im Editorbereich wird dabei zunächst ein einzelnes Objekt dargestellt, welches das Gesamtprojekt repräsentiert. Hier kann als Erstes ein Projekttitel und der momentane Projektstatus angegeben werden. Durch Rechtsclick auf das Projekticon öffnet sich ein Eigenschaftenfenster, in welchem weitere Einstellungen vorgenommen werden können. Wichtig hierbei sind vor allem die Eingabe eines Start- und Enddatums für das Projekt, deren Eingabe überprüft wird. Weiterhin sollten auf der Registerkarte „*Mitglieder*“ am Projekt beteiligte Benutzer zugeordnet werden.

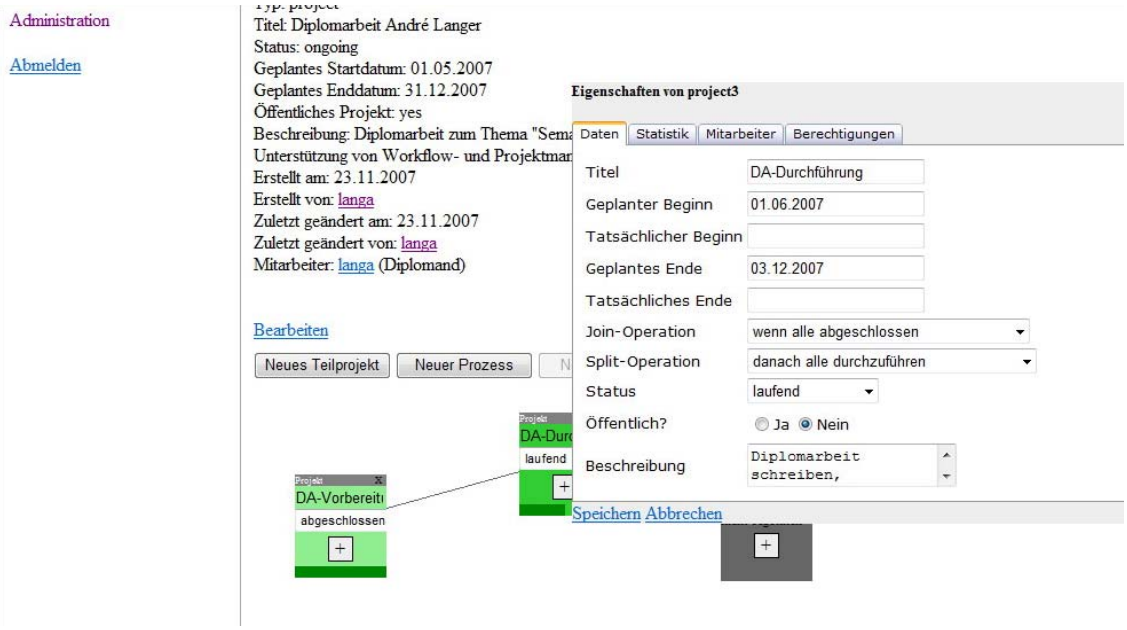


Abbildung 27: Projekteditor mit geöffnetem Eigenschaftenfenster

Es können nur dem System bekannte Benutzer ausgewählt werden, denen eine entsprechende Projektrolle zugeordnet werden kann. Anhand dieser Projektrollen können mehrere Personen gruppiert werden. Auf der Registerkarte „*Berechtigungen*“ können diese Gruppierungen dazu benutzt werden, mehreren Nutzern oder einzelnen Mitarbeitern Zugriffsrechte auf das Projekt zu gewähren. Durch Klick auf die Schaltfläche „*Speichern*“ kann das Eigenschaftenfenster geschlossen werden. Die Auswahl des Detailsymbols veranlasst den Projekteditor nun, in die nächste Detailebene des Projektes zu wechseln. Hier können zunächst weitere Unterprojekte oder Prozesse hinzugefügt werden, welche per Drag&Drop beliebig auf der Arbeitsfläche angeordnet werden können. Bei jedem Objekt empfiehlt es sich, im dazugehörigen Eigenschaftenfenster weitere Angaben zu machen. Standardmäßig erbt jedes Unterprojekt die Mitarbeiterangaben des übergeordneten Projekts. Ebenso darf der Zeitraum eines untergeordneten Teilprojektes nicht außerhalb des Gesamtprojektzeitraumes liegen. Zwischen zwei verschiedenen Projekten oder Prozessen können weiterhin Abhängigkeiten durch Klick auf die Schaltfläche („*Übergang definieren*“) definiert werden, indem anschließend per Klick beide Objekte markiert werden. Daraufhin wird eine gerichtete Kante eingefügt, welche ähnlich anderer Prozessnotationen eine Ausführungsreihenfolge symbolisiert. Der beschriebene Ablauf kann iterativ fortgesetzt werden, indem weitere Prozesse oder elementare Aktivitäten hinzugefügt werden, solange bis eine ausreichende Abstraktion des zu modellierenden Projekts erreicht ist. Aktivitäten können abschließend über einen Ressourcenmanager zusätzlich Dokumente zugeordnet werden, welche entweder zusätzliche Informationen für das ausführende Teammitglied enthalten oder das Ergebnis einer Aktivität darstellen können. Bei den Dokumenten kann es sich dabei sowohl um Verweise auf externe Ressourcen (URLs), hochgeladene Dokumente oder um einfachen Text handeln.

5.4. Testfälle

Um die Funktionalität und Korrektheit des entwickelten Systems zu testen, wurde abschließend eine Reihe von Testdaten auf das Projektmanagementsystem angewandt. In erster Linie soll dabei die Einsetzbarkeit für praktische Anwendungsfälle im Mittelpunkt stehen. Klassische Benchmarks orientieren sich dem gegenüber vor allem an physisch messbaren Größen, um die Performanz oder Hochverfügbarkeit eines Projekt- oder Workflowmanagementsystems zu charakterisieren, bei deren Messbarkeit vor allem Datenbankzugriffsmetriken eine wesentliche Rolle spielen.[48]

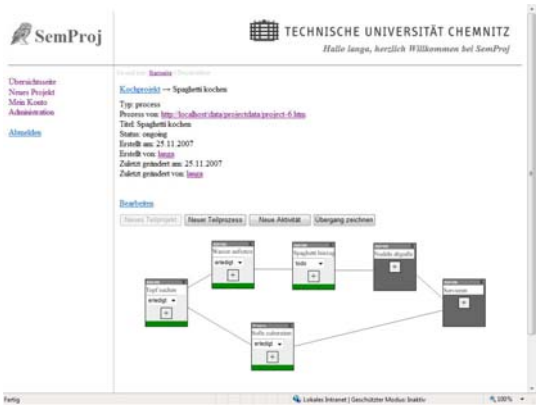
[48] Vgl. Benchmarking and Configuration of Workflow Management Systems, URL: http://www.mpi-inf.mpg.de/units/ag5/publications/benchmark-coopis.pdf

Anderorts werden Kriterienkataloge entwickelt, um ein Produktassessment[49] basierend auf den Eindrücken mehrerer Testpersonen durchzuführen oder Projektmanagementsysteme vergleichend gegenüberzustellen[50]. Diese kriterienbasierte Bewertung scheint drei wesentliche Nachteile in ihrer Anwendbarkeit auf das vorliegende System zu besitzen:

- Im Mittelpunkt steht häufig der Vergleich von Anwendungen aus dem B2B-Sektor unter Aspekten aus ökonomischer Sicht. Bei Applikationen, mit denen unternehmensfremde Prozesse, Aktivitäten aus dem Privatbereich oder abstraktere Tätigkeiten abgebildet werden soll, bringt eine Bewertung häufig unbefriedigende Ergebnisse
- Kriterienkataloge schaffen zwar eine Vergleichsbasis anhand objektiver Charakteristiken, entsprechen damit aber eher einer Evaluierung als einem Systemtest
- Verfügbare Benchmarks konzentrieren sich in der Mehrheit auf eine Managementkategorie. Systeme, welche sowohl Aspekte des Workflow- als auch des Projektmanagements abbilden können, werden darin kaum berücksichtigt.

Im Folgenden wird es als ausreichend erachtet, die Umsetzbarkeit der in 2.4. beschriebenen Anwendungsszenarien in dem entwickelten System zu testen. Deren Entwurf basierte auf der Zielsetzung, praktisch relevante Situationen mit zunehmender Komplexität abbilden zu können und ist daher für eine nähere Untersuchung geeignet.

a) Spaghetti kochen



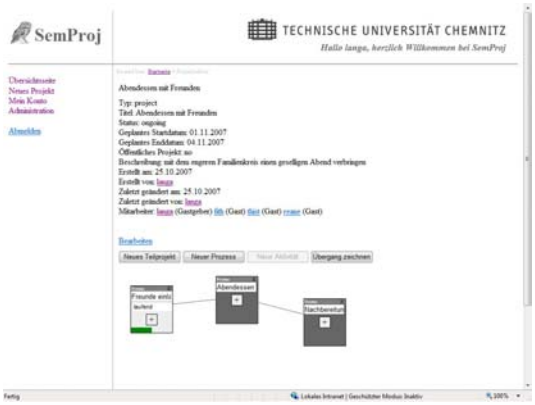
Um einen Workflow in der entwickelten Anwendung abbilden zu können ist es nötig, diesen in ein Projekt einzubetten. Um das Beispielszenario zu modellieren, gibt es verschiedene Möglichkeiten. So kann jeder Schritt des Workflows als einzelne Aktivität auf einer Hierarchieebene dargestellt werden, oder zunächst mit einer sehr groben Modellierung in Form einzelner Teilprozesse begonnen werden, welche schrittweise verfeinert werden (Prozess *Spaghetti kochen* besteht aus Prozess *Nudeln kochen* und Prozess *Soße zubereiten*). Zwischen den einzelnen Aktivitäten können Kanten eingefügt werden, wie in Abbildung dargestellt wird. Dabei werden mehrfache Split- und Join-Beziehungen unterstützt. Den Fortschritt des Workflows und die aktuell darin anstehenden Aufgaben zu erfassen ist einfach, da das Projektmanagementsystem alle Aktivitäten ausgraut, die noch nicht ausgeführt werden können.

[49] Siehe Fachgruppe der Deutschen Gesellschaft für Projektmanagement e.V., http://www.gpm-ipma.de/docs/showsite.php?menu=0102040121

[50] Ein umfangreicher Kriterienkatalog findet sich diesbezüglich unter http://www.pm-software.info/kriterienkatalog.html

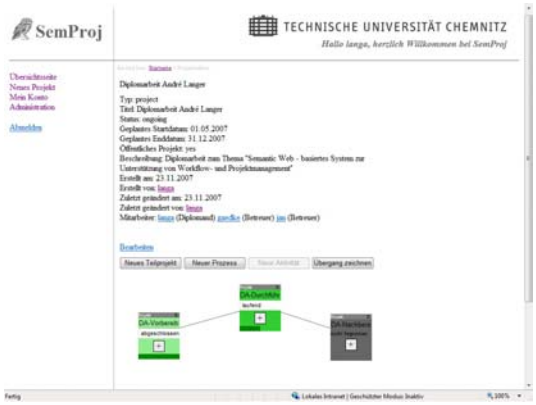
Sollten zur Modellierung weitere Unterprozesse verwendet werden, so wird der Arbeitsfortschritt innerhalb der Prozesse ebenfalls durch einen grafischen Fortschrittsbalken am unteren Rand des Prozessobjektes dargestellt. Die Erledigung einer Aufgabe kann direkt in der grafischen Repräsentation vermerkt werden (oder alternativ auf der Übersichtsseite), was bei den Testnutzern als eingängig und intuitiv bedienbar empfunden wurde. Bis auf die Notwendigkeit, zur Definition eines Workflows ein separates Projekt anlegen zu müssen, konnte Anwendungsszenario 1 problemlos mit dem semantischen Projektmanagementsystem umgesetzt werden.

b) Abendessen mit Freunden



Die Motivation hinter dem zweiten Anwendungsszenario war, bestehende Prozessdefinitionen in neuen Projekten wieder verwenden zu können. *SemProj* bietet dazu die Möglichkeit, nach bereits vorhandenen Projekt- oder Prozessbeschreibungen automatisch suchen zu lassen und bei Auswahl eines entsprechenden Objektes alle enthaltenen Unterdefinitionen zu kopieren und als Unterelemente in das neu erstellte Projekt einzufügen. Diese Importfunktion funktionierte, auch wenn sie qualitativ verbessert werden könnte. So wird die ursprüngliche Datei der zu importierenden Beschreibung kopiert und darin enthaltene spezifische Beschreibungen wie der aktuelle Ausführungsstatus oder Zugriffsrechte werden zurückgesetzt. Dies stellte sich in einem Testlauf als nachteilig hinaus, da Zugriffsrechte für Projektbeteiligte im neuen Kontext von Hand neu gesetzt werden müssen. Ansonsten wurde die Benutzer- und Rollenzuordnung als sinnvoll nutzbar angesehen, da sie eine sehr flexible Zugriffsbeschreibung zulässt, welche sich an real existierenden Bezeichnungen orientiert und keine festen Rollen vorgibt, wie dies in anderen Projektmanagementsystemen häufig zu finden ist. Zur Modellierung des Anwendungsszenarios konnten so als Projektbeteiligte zunächst Benutzer in der Rolle des *Gastgeber*s definiert und andere Benutzer in der Rolle eines *Gast*es gruppiert werden. In einem Unterprozess wie dem von Anwendungsszenario eins besteht im Weiteren die Möglichkeit, den am Projekt beteiligten Benutzern neue Rollen zu geben. So könnte ein Benutzer in der Rolle des *Gastgeber*s in einem Teilprojekt beispielsweise die Rolle *Koch* annehmen, was eine natürlichsprachliche Modellierung fördert. Momentan besitzt die Umbenennung von Rollenzuordnungen keine effektive Nutzungsmöglichkeit, da sich deswegen noch keine Zugriffsberechtigungen ändern müssen (ob der gleiche Nutzer als *Gastgeber* oder *Koch* bezeichnet wird, ist dafür nicht relevant), für eine Weiterentwicklung des Systems könnte dieses Konzept aber weiter ausgebaut werden.

c) Diplomarbeit schreiben



Das Szenario, mit dem Projektmanagementsystem die Anfertigung einer Diplomarbeit darzustellen und deren Ausführung in Form eines Workflows zu überwachen, wurde bereits in den vorausgehenden Kapiteln mehrfach angeführt. In einem Testlauf stellt sich diese Anwendung als problemlos umsetzbar heraus. So werden zunächst Projektbeteiligte definiert, welche in die Rollen *Diplomand, Betreuer* und ggf. *externe Mitarbeiter* (bspw. aus dem Prüfungsamt) eingeteilt werden. Trotz dass eine explizite Darstellung fehlt, welche Person für welche Aufgabenausführung zuständig ist (Swimlane-Ansatz bei BPMN), kann ein Testbenutzer direkt erfassen, welche Aufgaben in dem Projekt als Nächstes anstehen. Auf der Projektübersichtsseite findet sich weiterhin eine Aufschlüsselung, ob der Benutzer selbst die nächste Aktivität auszuführen hat oder ob zunächst Aktionen durch andere Nutzer ausgeführt werden müssen. Der Fakt, dass der Anfertigungsprozess einer Diplomarbeit nicht in seiner Gesamtheit dargestellt wird (wie beispielweise in einem BPMN-Graphen), sondern nur ausschnittsweise anhand einzelner Projekt- oder Prozessebenen, macht sich bei einem ersten Testlauf als durchaus positiv bemerkbar. Patterns aus dem Bereich der Medieninformatik bezüglich der Rezeption von Informationen können darin wiedergefunden werden, nach denen ein Nutzer optimal bis zu fünf dargestellte Fakten (chunks) auf einer einzelnen Seite aufnehmen kann, darüber hinaus aber leicht überfordert ist und die Darstellung an Übersichtlichkeit verlieren würde. Der Ansatz, ein sehr einfach gehaltenes Projektmanagementsystem zu entwerfen, zeigt sich hier als durchaus positiv. Ebenso reichen für dieses Anwendungsszenario weitere bereitgestellte Funktionen aus, mithilfe deren einfach Dokumente zugeordnet und verwaltet werden können. So besteht für einen *Betreuer* direkt die Möglichkeit zu erkennen, ob der *Diplomand* eine *neue Version der Diplomarbeit hochgeladen* hat. Ebenso hat der *Diplomand* die Möglichkeit, *überarbeitete Versionen* durch den *Betreuer* abzurufen oder sonstige Formalitäten zu regeln.

d) Webseiten- und Applikationsdesign

Prinzipiell können mit dem vorliegenden Projektmanagementsystem ebenso komplexere Anwendungen aus dem Unternehmensalltag abgebildet werden. Ein Entwicklungsprojekt kann in einem ersten Verfeinerungsschritt zunächst in projekttypische Phasen wie *Analyse, Design, Implementierung* und *Test* zerlegt werden. Gleichzeitig sind parallele Entwicklungen modellierbar, sodass z.B. eine Spezifikation begleitend ständig aktualisiert werden muss oder *Programmierer* und *Layouter* nebeneinander her arbeiten.

Die Zuordnung einzelner Mitarbeiter zu Projektrollen stellte sich in einem Testlauf als unkompliziert durchführbar heraus, allerdings zeigte sich ebenso, dass in dem Prototypen einige wesentliche Funktionen für einen praktischen Einsatz noch fehlen. Exemplarisch seien Möglichkeiten genannt, einzelne Workflows inkrementell in mehreren Durchläufen durchführen zu können. Ebenso existiert momentan noch keine Möglichkeit, die Weiterausführung eines Workflows von der Bestätigung mehrerer Benutzer abhängig zu machen, wie sich dies in vielen praxisrelevanten Workflowmanagementsystemen wieder findet.

e) Workshoporganisation

Das komplexeste Anwendungsszenario, einen Workshop zu organisieren, an deren Planung und Vorbereitung ein ganzes Team über mehrere Wochen beteiligt ist, konnte in einem Testlauf mit der vorliegenden Version des *SemProj*-Systems nicht befriedigend umgesetzt werden. Ein Grund dafür war, dass dazu eine Vielzahl an Teilprojekten, Prozessen und Einzelaktivitäten zu definieren ist, um alle Zuständigkeiten und zu erledigenden Aufgaben zu erfassen. Da dem Projektmanagementeditor eine hierarchische Entwicklungsstruktur zugrunde liegt, wuchs die Anzahl an durch das Projektmanagementsystem verwalteten Objektbeschreibungen sehr schnell. Als Folge davon stiegen die Verarbeitungszeiten bezüglich der Extraktion und Auswertung der in den Dateien enthaltenen Metadaten schnell an, wodurch die Einrichtung des Projektes zu einer langwierigen Tätigkeit wurde und nach mehreren Schritten entweder abgebrochen wurde oder auf eine feingranularere Darstellung einzelner Prozesse verzichtet werden musste und stattdessen nur einfache textuelle Beschreibungen einzelner Prozesse verwendet wurden, was nicht Sinn eines Projekt- oder Workflowmanagementsystems sein sollte. Bei entsprechender Weiterentwicklung könnte jedoch auch dieser Anwendungsfall praktisch nutzbar modelliert werden, da das Projektmanagementsystem dazu an sich flexibel genug ist. Lediglich der Entwicklungszeitraum von wenigen Wochen schien dafür nicht ausreichend, alle Funktionen so ausgereift zu implementieren, dass diese lange Jahre im Einsatz befindlichen Managementsystemen in nichts nachstehen.

6. Diskussion

„The future belongs to those who see possibilities before they become obvious." ~ Samuel Johnson

6.1. Beurteilung des Systementwurfs

Die Beschreibung des Systems aus Benutzersicht („Benutzerhandbuch") in Kapitel 5.3. kam ohne einen einzelnen Verweis auf die dahinter liegende, semantisch annotierte Datenbasis aus. Dieser Sachverhalt, dass ein Endanwender bei entsprechend sauberer Implementierung von RDFa, Microformats oder einem beliebigen anderen Annotationsformat keinerlei Kenntnis benötigt und sich aus Nutzersicht an der Bedienung des Systems nichts ändert, soll hierbei explizit herausgestellt werden. Dies verwundert nicht, da der Nutzer die Inhalte einer Webseite sowohl mit als auch ohne semantischen Metainformationen intuitiv erfassen kann. Ziel eines Semantic Web ist es jedoch, diese Informationen auch in einer für Maschinen verständlichen Repräsentationsform bereitzustellen (machine-machine-interaction). Für den Endanwender ist dieser Prozess weitestgehend transparent (es wäre sogar extrem nachteilig, wenn der menschliche Benutzer in einem semantikbasierten System nur noch Ein- und Ausgaben in Form einzelner Tripel tätigen müsste). Dennoch ist von einer technischen Neuentwicklung ein gewisser Mehrwert zu erwarten. Dieser findet sich in einem semantikorientierten System in einer Vielzahl an kleinen Funktionen wieder, welche dem Nutzer qualitativ bessere Ergebnisse liefern oder welche bewusst gar nicht einmal wahrgenommen werden. Entsprechend finden sich Parallelen zu anderen Technologien wie AJAX (Asynchronous Javascript and XML), womit reichhaltigere Funktionalitäten bereitgestellt werden können und dies bei einem Nutzer einen positiveren Gesamteindruck hinterlassen kann.

In der Vision des Semantic Web spielen Suchagenten eine wesentliche Rolle. Häufig werden Vergleiche mit heutigen Suchmaschinen angestellt, welche zukünftig nur noch auf eine Anfrage passende Ergebnisse zurückliefern können. In dem in dieser Arbeit realisierten Projektmanagementsystem findet sich eine ähnliche Funktion, bei der eine Liste bereits existierender Projekte und Prozesse zurückgegeben wird, deren Titel mit der durch den Nutzer eingegebenen Zeichenkette beginnt. Die Auflistung basiert dabei nicht alleinig auf einem Zeichenkettenvergleich sondern enthält nur Elemente, welche den gleichen Typ besitzen wie das Objekt, in dem die Suche initiiert wurde. Ein weiteres Beispiel für das Potential in semantischen Suchagenten kann in der Projektübersichtsseite gesehen werden. Statt nur eine Liste mit Verweisen auf alle existierenden Projekte anzubieten, filtert *SemProj* die gefundenen Ergebnisse nach Kriterien entsprechend dem angemeldeten Nutzer, der diese Seite aufruft und nach dem Status der laufenden Projekte, Prozesse und Aktivitäten.

Eine kleine Workflow-Engine aggregiert anschließend die Informationen aus verschiedenen Internetressourcen in einer kompakten Übersicht, sodass der Nutzer auf der Übersichtsseite nicht nur eine Auflistung der durchzuführenden Aufgaben vorfindet, sondern gleichzeitig eine kompakte Darstellung, zu welchem Projekt diese Aktivitäten gehören, und in welchem Prozess diese zu finden sind. Bisher hätte der Nutzer in diesem Fall wahrscheinlich die Dateien einzeln aufrufen und sich selbst von einer Ressource zur nächsten verlinkten Ressource weiterbewegen müssen, um alle entsprechenden Informationen zu erhalten. Die Idee von intelligenten Agenten, welche dem Menschen mühsame Arbeiten abnehmen und bei alltäglichen Aufgaben unterstützen können, scheint in diesem Zusammenhang keine Utopie mehr zu sein. Gleichzeitig ist anzumerken, dass dies prinzipiell auch ohne eine mit Semantikinformationen versehene Datenbasis realisierbar ist. Daten mithilfe von Computeralgorithmen vergleichen, manipulieren und zusammenfassen zu können, ist an sich nichts Neues und in gleicher Art von jedem anderen Projekt- oder Workflowmanagementsystem realisierbar. Der wesentliche Unterschied besteht darin, in diese Algorithmen nun automatisiert Inferenzoperationen basierend auf dahinterliegenden Klasseninformationen einschließen zu können, was über eine bisherige Verarbeitung hinausgeht, welche sich auf syntaktischen Informationen über Datentypen und Datenstrukturen stützte. Weiterhin wird eine Informationsverarbeitung institutionsübergreifend ermöglicht, ohne dass dazu Austauschformate mit einem speziellen Schema oder explizite Zugriffspunkte nötig wären. Auch dieser Ansatz wurde in der vorliegenden Implementierung untersucht, indem bereits in der Analysephase eine verteilte Bereitstellung der Projektdaten in das Konzept einbezogen wurde. Statt alle Informationen über ein Projekt oder einen Workflow mit allen Unterelementen in einer einzelnen Datei zu verwalten, wie dies bei anderen Workflowbeschreibungssprachen typisch ist, werden hier die Daten jedes einzelne Objektes in einer einzelnen Datei („Ressource“) hinterlegt, welche direkt über eine URL adressierbar ist und so flexibler mit den Informationen aus anderen Dateien kombiniert werden kann (die Hinterlegung von Informationen über mehrere Objekte in einer Datei ist aber dennoch möglich). Dies entspricht der Beschreibung des Semantic Web als eine Art riesige, verteilte Datenbank im Internet, aus der unterschiedlichste Anwendungen beliebige Informationen auslesen können, ohne spezielles Hintergrundwissen über die Datenrepräsentation von der „Insel“ zu besitzen, der diese Informationen entstammen. In diesem Zusammenhang wird gleichzeitig ein Problem des semantischen Internets deutlich: Es setzt auf Kollaboration, dies bedeutet gegenseitige Zusammenarbeit und Vertrauen. Das Problem ist nicht, dass die Auszeichnung von Informationen anhand deren Bedeutung aufwändig wäre, sondern dass der Zugriff auf diese Information weltumspannend ermöglicht wird. Der Aufwand, den ein Unternehmen oder eine Einzelperson betreibt, um Daten mit semantischen Metainformationen bereitzustellen, kann nur damit gerechtfertig werden, dass 1. dadurch entweder diese Daten effektiver weiterverwendet werden können oder man sich 2. erhofft, dass andere Endanwender genau den gleichen Aufwand betreiben und man dafür Informationen aus deren Wissensbasis zu eigenen Zwecken weiterverwenden kann.

Bei der Betrachtung eines Projektmanagementsystems mag dieser Schritt kritisch sein, da vielfach mit sensiblen Daten umgegangen wird, deren Kenntnis für konkurrierende Anbieter von großem Wert sein könnte. In der Systemarchitektur herrscht daraus resultierend ein Widerspruch, dass einerseits nur autorisierten Benutzern der Zugriff auf die Daten gestattet werden soll, andererseits das Semantic Web das Interesse verfolgt, möglichst viele Informationen derart bereitzustellen, dass sie von externen Applikationen gefunden und weiterverarbeitet werden können. Der scheinbare Widerspruch kann dahingehend aufgelöst werden, dass heutzutage bereits eine Vielzahl an kommerziellen Anwendungen lokal RDF-annotierte Daten verarbeitet, welche diese Metainformationen zur besseren Analysierbarkeit und Verwaltbarkeit halten ohne dass diese in einem globalen Kontext weiterverwendet werden. Semantische Beschreibungen sind damit zur maschinellen Verarbeitung bereits so nützlich, dass damit die Systemarchitektur eines Projektmanagementsystems rechtfertigbar ist. Wird dieses Projektmanagementsystem als Webapplikation bereitgestellt, so ist die Datenintegrität und eine Zugriffsbeschränkung auf die Daten in besonderer Weise sicherzustellen. Im vorliegenden System werden Informationen aus den durch das Projektmanagementsystem verwalteten Dateien mithilfe einzelner Web Requests gelesen. Prinzipiell wäre zwar auch ein Zugriff auf Dateiebene denkbar, doch würde dies die Anwendung stark auf die eigenen Systemgrenzen beschränken. Sinnvoller ist es gewesen, den Zugriff auch auf entfernte Ressourcen kooperativer Systeme zuzulassen. Dies sollte auch möglich sein, ohne dass diese einen dedizierten Webservice bereitstellen (analog dem Beispiel, dass in Zukunft beliebige Suchagenten das Semantic Web nach Informationen durchsuchen). Änderungen an den bestehenden Daten sind aber nur durch Zugriff auf bereitgestellte Methoden eines SemProjWebservices möglich. Der lesende Zugriff auf Webdokumente sollte in einem praktischen Einsatz des Systems dennoch eingeschränkt werden. Die einfachste Möglichkeit wäre, nur Web Requests zuzulassen, die dem Application Server (localhost) entstammen. Das Identity Federation System (idFS) bietet darüber hinaus fortgeschrittene Möglichkeiten. Der Security-Aspekt wurde in der Systemarchitektur nur punktuell behandelt, stellt sich für weitere Betrachtungen aber als essentiell wichtig dar.

Im Zusammenhang mit der Verwaltung verteilter Informationen im realisierten Projektmanagementsystem wurde jedoch auch ein wesentlicher Schwachpunkt identifiziert. In der ursprünglichen Systemarchitektur wurde sich dazu entschieden, sowohl die eigentlichen Projektdaten als auch die dazugehörigen (RDFa/Microformat-) Metainformationen in einer physisch existenten Datei zu halten und zu verwalten. Dies wurde damit begründet, dass das System das Verhalten eines Suchagenten simulieren könnte, der letztendlich auch nur existierende Webdokumente aufrufen und deren Inhalt verarbeiten kann und kein Wissen darüber besitzt, ob diese nun dynamisch erstellt worden oder statische HTML-Dateien sind. Einzelne Meinungen im Forschungsbereich gingen weiter und warfen die Frage auf, ob es überhaupt sinnvoll sei, Daten dynamisch aus einer Datenbank zu extrahieren und diese in einem zweiten Schritt mit Metainformationen zu versehen, wo als Resultat dessen wieder eine Art Datenbank entstehen würde.

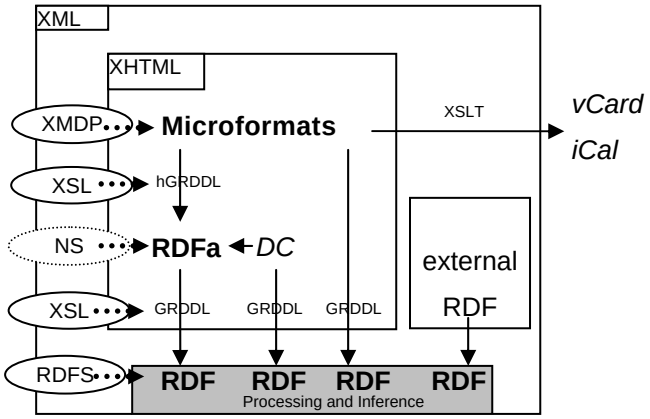


Abbildung 28: Transformation semantischer Metainformationen nach RDF

Den Aufwand könne man sich sparen und alle Daten direkt auf Dateisystemebene halten. Diesem Ansatz kann rückblickend nicht zugestimmt werden, da dadurch mehrfach der Implementierungsprozess behindert oder unnötig verlängert wurde und der Aufwand zusammenfassend nicht gerechtfertigt ist. Bei einer Weiterentwicklung oder Neuimplementierung des *SemProj*-Systems würde die Systemarchitektur so modifiziert werden, dass Dokumentinhalte dynamisch aus einem dahinterliegenden, unabhängigen Repository generiert werden. Damit können die Vorteile von Datenbankmanagementsystemen effektiv eingesetzt werden, welche in den zurückliegenden Jahrzehnten entwickelt und sich bewährt haben und den Programmierer von dem Prozess der manuellen Datenhaltung weitestgehend entbinden. Dieser Vorgehensweise steht in einem semantikbasierten System nichts entgegen sondern wird vielmehr noch dadurch gefördert, dass sich das RDF-Format als sehr flexibles Austauschformat zwischen heterogenen Systemen etabliert zu scheinen hat, wie Abbildung 28 zeigt. Dabei ist es für eine maschinelle Verarbeitung der Informationen als RDF statements unerheblich, ob die Daten ursprünglich einer HTML-Datei, XML-Datei, Datenbank oder einem anderen beliebigen RDF Repository entstammen.

Während der Implementierungsphase ist ein weiterer Vorteil bei der semantischen Auszeichnung von Dokumentinhalten aufgefallen. Da darüber Daten in Klassen zusammengefasst werden und diesen Klassen (Konzepten) charakteristische Eigenschaften und Beziehungen zugeschrieben werden können, wird eine direkte Verbindung zwischen der Verwaltung einzelner Daten und der programmatischen Verarbeitung im Sinne der objektorientierten Programmierung hergestellt. Die objektorientierte Denkweise wird gefördert und kann bereits zu Beginn Widersprüchlichkeiten in der Datenrepräsentation aufdecken. Ebenso können Informationen in natürlichsprachlicherer Art und Weise abgebildet und verwaltet werden, wodurch der Implementierungsprozess unterstützt wird. Konzepte einer Ontologie können anschließend direkt in Klassenbeschreibungen überführt werden.

Da bei dem zu entwickelnden System besonderer Schwerpunkt auf die Integration semantischer Informationen in bestehende XHTML-Dokumente gelegt wurde, wirft dies eine weitere Frage auf. Während, wie in Kapitel 1 beschrieben, HTML in erster Linie zur Auszeichnung von Darstellungsinformationen verwendet wird, werden XML-Dokumente parallel dazu seit langem zur Datenrepräsentation und zum Datenaustausch verwendet. Durch die Integration semantischer Informationen in normale XHTML-Dokumente scheint diese strikte Trennung interessanterweise wieder zu verschwinden. In dem entwickelten System stellt sich dieser Trend als nicht nachteilig heraus. Es wird angenommen, dass es aufwändiger gewesen wäre, parallel zu den eigentlich dargestellten Informationen in einer HTML-Datei deren Bedeutung in einer separaten XML-Datei mit RDF statements führen zu müssen, da diese Redundanz bei jedem Änderungsvorgang aufwändige Aktualisierungsoperationen nach sich gezogen hätte um die Integrität der Daten sicherzustellen (abgesehen von dem zusätzlichen Speicherplatzverbrauch).

6.2. Vergleich zu bisherigen Managementsystemen

Bisher wurden Vor- und Nachteile der zugrunde liegenden Systemarchitektur eines semantikbasierten Projektmanagementsystems diskutiert, doch ist noch die Frage offen geblieben, inwieweit sich ein derartiger Ansatz von gängigen Managementapplikationen unterscheidet, welche in Kapitel 2.5. bereits untersucht wurden. In Abschnitt 6.1. wurde bereits die Aussage getroffen, dass sich das Bedienkonzept und das Nutzerinterface von „herkömmlichen“ Projektmanagementsystemen und einem semantikgestützten Projektmanagementsystem nicht wesentlich voneinander unterscheidet.

Vorteilhaft stellen sich semantikorientierte Systeme heraus, wenn Daten aus unterschiedlichen Quellen kombiniert werden und anschließend in einem anderen Format dargestellt werden sollen. Im Projektmanagementbereich ist beispielsweise die Darstellung in Form von Gantt-Diagrammen weit verbreitet. Ebenso kann der Projektverlauf in eine Kalenderdarstellung umgewandelt werden, in der alle geplanten und realen Termine von Teilprojekten (Meilensteinen), Prozessen und Einzelaktivitäten eingetragen sind. Prinzipiell können beide Darstellungsformen sowohl aus einer herkömmlichen Datenbank als auch aus einer verteilten Datenbasis gewonnen werden. Gelingt es, Informationen aus Projekt- und Prozessbeschreibungen automatisiert in eine Kalenderdarstellung zu überführen, sind weitere Operationen auf dieser neuen Darstellungsform möglich. So wäre automatisch eine Dokumentation vorhanden, zu welchem Zeitpunkt ein Projektteilnehmer eine konkrete Aufgabe erfüllt, oder beispielsweise ein Treffen stattgefunden hat. An diesem Punkt spielen die Microformat-Annotationen eine wichtige Rolle, die zur bisherigen Systemfunktionalität nur wenig beigetragen haben. Viele Projektmanagementsysteme bieten zwar eine Kalenderfunktion, welche jedoch entweder keine Exportfunktion, sondern nur eine Benachrichtigungsmöglichkeit für Termine anbietet, oder die Termindaten zunächst in ein anderes Format umwandelt, welches durch den Nutzer herunter geladen und in einer anderen Anwendung wieder geladen werden muss.

Microformat-bewusste Anwendungen könnten dem gegenüber alle Termine sofort und ohne Aufwand zur Weiterverwendung bereitstellen. Ein ähnliches Vorgehen wäre bei der Weiterverwendung von Kontaktdaten der an einem Projekt beteiligen Nutzer möglich.

Ebenso können Darstellungen in anderen Formaten abgeleitet werden. Dies ist dahingehend interessant, dass Daten, welche ursprünglich als Projektbeschreibungen in verschiedenen Dateien vorhanden waren, in einer Datei zusammengefasst werden können, welche alle Informationen gebündelt enthält. Im Workflowmanagement sind diesbezüglich XPDL-Beschreibungen (XML Process Definition Language)[51] sehr weit verbreitet. Da die Projekt- und Prozessontologien domainabhängig entworfen wurden und sich andere Beschreibungen ebenfalls an charakteristischen Eigenschaften dieser Konzepte orientieren, wäre die Generierung einer einfachen XPDL-Beschreibung basierend auf vorhandenen Informationen eines Projektes und dessen Unterstruktur problemlos möglich. So finden sich in XPDL viele der Konzepte wieder, welche bereits in Abschnitt 3.6. angesprochen worden (*Process, Activity, Transition, Participant* – vgl. Abbildung 29).

Package	Workflow Process	Activity	Transition	Application	Data Field (Workflow Relevant Data)	Participant
- Id - Name - Description - Extended Attributes - XPDL Version - Source Vendor ID - Creation Date - Version - Author - Codepage - Country Key - Publication Status - Conformance Class - Priority Unit	- Id - Name - Description - Extended Attributes - Creation Date - Version - Author - Codepage - Country Key - Publication Status - Priority - Limit - Valid From Date - Valid To Date	- Id - Name - Description - Extended Attributes - Automation Mode - Split - Join - Priority - Limit - Start Mode - Finish Mode - Deadline	- Id - Name - Description - Extended Attributes	- Id - Name - Description - Extended Attributes	- Id - Name - Description - Extended Attributes - Data Type	- Id - Name - Description - Extended Attributes - Participant Type
- Responsible - External Package	- Parameters - Responsible	- Performer - Tool - Subflow - ActivitySet - Actual Parameter	- Condition - From - To	- Parameters	- Initial Value	
- Documentation - Icon	- Documentation - Icon	-Documentation- -Icon				
- Cost Unit	-Duration Unit - Duration - Waiting Time - Working Time	-Cost - Duration - Waiting Time - Working Time				

Abbildung 29: Attribute einer XPDL-Beschreibung

[51] Vgl. Spezifikation unter http://www.wfmc.org/standards/docs/TC-1025_xpdl_2_2005-10-03.pdf mit XML Schema Definition unter http://www.wfmc.org/standards/docs/TC-1025_bpmnxpdl_24.xsd

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<Package xmlns="http://www.wfmc.org/2002/XPDL1.0" xmlns:xpdl="http://www.wfmc.org/2002/XPDL1.0" xmlns:xsi=
"http://www.w3.org/2001/XMLSchema-instance" Id="Spaghetti_kochen" Name="Spaghetti kochen" xsi:schemaLocation=
"http://www.wfmc.org/2002/XPDL1.0 http://wfmc.org/standards/docs/TC-1025_schema_10_xpdl.xsd">
    <PackageHeader>
        <XPDLVersion>1.0</XPDLVersion>
        <Created>2007-09-04 18:44:06</Created>
    </PackageHeader>
    <Participants>
        <Participant Id="person1" Name="Andre">
            <ParticipantType Type="HUMAN"/>
        </Participant>
    </Participants>
    <WorkflowProcesses>
        <WorkflowProcess AccessLevel="PUBLIC" Id="Spaghetti_kochen_wp1" Name="Spaghetti_kochen_wp1">
            <ProcessHeader>
                <Created>2007-09-04 18:52:37</Created>
            </ProcessHeader>
            <Activities>
                <Activity Id="Spaghetti_kochen_wp1_act1" Name="Wasser aufsetzen">
                    <Implementation>
                        <No/>
                    </Implementation>
                    <Performer>person1</Performer>
                    <StartMode>
                        <Manual/>
                    </StartMode>
                </Activity>
                <Activity Id="Spaghetti_kochen_wp1_act2" Name="Salz hinzugeben">
                    <Implementation>
                        <No/>
                    </Implementation>
                    <Performer>person1</Performer>
                </Activity>
                <Activity Id="Spaghetti_kochen_wp1_act3" Name="Spaghettis hinzufügen">
                    <Implementation>
                        <No/>
                    </Implementation>
                    <Performer>person1</Performer>
                </Activity>
```

Abbildung 30: "Spaghetti kochen" als XPDL-Beschreibung

Hauptanwendung von XPDL ist die Beschreibung von Workflows und der Austausch von Prozessbeschreibungen zwischen verschiedenen Workflow-Engines; von daher sollte das Konzept des Projektmanagements eine eher untergeordnete Rolle spielen. Dem gegenüber ist in XPDL jedoch auch eine Packagebeschreibung zu finden, welche im engeren Sinne einen Projektcontainer spezifiziert, in dessen Kontext die enthaltenen Workflows zur Ausführung kommen. Daraus ist ersichtlich, dass eine semantikgestützte Applikation im Prinzip mit jeder beliebigen anderen Anwendung zusammenarbeiten kann, ohne dass ein proprietäres Zwischenformat erforderlich ist, welches die Kooperation eines Projektmanagementsystems mit einem Workflowmanagementsystem ermöglicht. Das beispielhafte Transformationsergebnis des Anwendungsszenarios 1 („*Spaghetti kochen*“) in eine XPDL-Beschreibung ist ausschnittsweise in Abbildung 30 dargestellt.

Die Wiederverwendung von Projekt- und Prozessbeschreibungen durch eine semantikbewusste Webapplikation ist nicht auf durch das Projektmanagement von Anfang an verwalteten Ressourcen beschränkt. Stelle man sich andere Webseitenautoren im Semantic Web vor, welche ihre Inhalte mit einer ähnlichen semantischen Auszeichnung versehen, so könnten auch diese Informationen aus den entsprechenden Webseiten extrahiert und in der eigenen Webseite weiterverwendet werden. Beispielsweise könnte eine Internetseite existieren, auf der eine Illustration zu finden ist, wie eine Diplomarbeit geschrieben werden sollte. Obwohl auf der Webseite der Sachverhalt durch eine Illustration mit mehreren Bildern zu finden ist, könnte das Projektmanagementsystem bei entsprechend vorhandener Annotation der Grafiken eine Prozessbeschreibung ableiten, welche in das eigene Projekt eingegliedert werden kann.

Der Autor der Webseite muss dazu nicht einmal die gleiche Projektmanagement-Ontologie verwenden, da nach den Vorstellungen des Semantic Web Übersetzungswerkzeuge existieren können, welche die unterschiedliche Beschreibung gleicher Konzepte in verschiedenen Ontologien ineinander überführen können.

Abschließend sei bemerkt, dass trotz der bisherigen Darstellung eine auf RDFa- oder Microformats-basierende Anwendung gegenüber herkömmlichen Applikationen nicht ausschließlich Vorteile bietet. Wie in Kapitel 5.2. beschrieben, ist der Extraktionsaufwand von RDF statements aus dem Quellcode einer Internetseite nicht zu unterschätzen, insbesondere wenn eine Reihe von Dokumenten zur Informationsgewinnung herangezogen werden muss. Ebenso sind Daten anhand einer Konzeptzuordnung zwar maschinell verarbeitbar, was aber nicht bedeutet, dass sich dadurch der Entwicklungsaufwand verringert oder eine programmatische Verarbeitung gar überflüssig werden würde. Anhand des entwickelten Prototypen konnte gesehen werden, dass die Entwicklung eines „Semantic Web“-basierten Systems durchaus aufwändig werden kann. Ähnlich wie bei anderen, beispielsweise AJAX-gestützten Anwendungen, liegt der Entwicklungsaufwand zunächst auf Programmiererseite, während es für den Nutzer letztendlich eine Vereinfachung und mehr Komfort bringen soll. Im Gegensatz dazu existiert bereits eine Reihe sehr leistungsfähiger, ausgereifter Webapplikationen im Bereich des Projekt- und Workflowmanagements, welche zwar mithilfe semantischer Annotationen (Microformats) Daten für eine zusätzliche Exportmöglichkeit bereitstellen könnten, ansonsten aber keine wesentliche qualitative Aufwertung durch eine semantisch annotierte Datenbasis erfahren würden. So gesehen bietet ein semantikbasiertes Projektmanagementsystem gegenüber konventionellen Webapplikationen zwar neue Funktionsmöglichkeiten, rechtfertigt aber (noch) nicht eine Neuimplementierung der vorhandenen Systeme oder könnte diese in Ihrem Anwendungsumfang übertreffen.

6.3. Weiterentwicklungsansätze

In der vorliegenden Arbeit wurde gezeigt, dass es inzwischen möglich ist, Webanwendungen zu entwickeln, welche die Beziehung zwischen darin verarbeiteten Ressourcen erkennen und zu weiteren Analyse- und Verarbeitungsoperationen nutzen können. Basis für derartige Operationen stellt das Ressource Description Framework dar. Ebenfalls wurde gezeigt, dass unter einer konkreten Problemstellung die problemlose Kombination verschiedener Ontologien möglich ist. Im Gegensatz zu der vielfach im Internet angeführten Nutzungsmöglichkeit von Ontologien zur ausschließlich Darstellung von vCards oder FOAF-Profilen konnte gezeigt werden, dass dieser Ansatz eine wesentlich größere Flexibilität für komplexere Anwendungen bietet.

Im Bezug auf eine mögliche Weiterentwicklung des semantischen Projektmanagementsystems stünde in erster Linie eine Verbesserung der Performance und Skalierbarkeit der Anwendung beim Zugriff auf eine Vielzahl verteilter Projekt- und Prozessdefinitionen im Mittelpunkt. Ebenso müssten für einen industriellen Einsatz einige Funktionen wie der Im- und Export von Projektbeschreibungen ausprogrammiert und in ihrer Korrektheit gegenüber vorhandenen Standards überprüft werden, welche in dem Prototypen momentan nur als Funktionsrümpfe zu Demonstrationszwecken in Form einer Machbarkeitsanalyse vorhanden sind.

Darüber hinaus ist eine Vielzahl weiterer Funktionen denkbar, welche in einem semantischen Projektmanagementsystem umgesetzt werden könnten, bisher aber noch nicht betrachtet wurden. Vorstellbar wäre unter anderem eine natürlichsprachliche Bereitstellung einer Workflowbeschreibung als Fließtext, in dem der Leser eine Schritt-für-Schritt-Anleitung zur Durchführung einer konkreten Aufgabe erhält. Die Informationen in dem Text könnten automatisch aus vorhandenen Projekt- und Prozessbeschreibungen abgeleitet und anhand von Semantik und Kontext in einem verständlichen Text kombiniert werden, beispielsweise so: *„Das Projekt Diplomarbeit besteht aus drei Teilen – der Themenfindung, der Umsetzung und der Verteidigung. Der erste Abschnitt Themenfindung besteht aus den Schritten Betreuer suchen, Treffen vereinbaren, Thema festlegen und Thema anmelden. Diese Aufgabe wurde von Ihnen bereits abgeschlossen […]“* Durch eine Annotation der im Text enthaltenen Aussagen mit semantischen Metainformationen könnte wiederum das Hypertext-Konzept genutzt werden, indem für den Leser unbekannte oder interessante Fakten verlinkt und in detailierter Form auf einer Unterseite genauer nachgelesen werden können. Das Ergebnis käme dem Einstiegsbeispiel eines intelligenten Handhelds aus Kapitel 1 bereits recht nah, umfasst neben einer Auswertung semantischer Informationen und Wissen über Projektmanagement eine Reihe weiterer Forschungsfelder wie beispielsweise die Verarbeitung natürlicher Sprache bezüglich grammatikalischer Richtigkeit.

Ebenso könnten die bisher nur rudimentär vorhandenen Loggingmöglichkeiten des Projektmanagementsystems weiter ausgebaut werden, indem zu jedem Zeitpunkt nachvollziehbar ist, welcher Nutzer zu welchem Zeitpunkt welche Aktion in den Systemgrenzen ausgeführt hat. Unter einem semantischen Gesichtspunkt könnten dazu weitere Informationen aus externen Datenquellen mit einbezogen werden. So wäre es in einigen Anwendungsszenarien interessant zu wissen, wann ein Projektbeteiligter einem anderen Nutzer eine Email geschickt hat und ob dieser Nutzer auf diese Mail geantwortet hat. Seit einiger Zeit existieren dazu im World Wide Web Email Tracking Anbieter, deren Konzept prinzipiell dafür genutzt werden könnte. Ebenso können Informationen aus anderen Anwendungen interessant sein (beispielsweise ob und wann eine neue Programmversion in ein SVN Repository eingecheckt wurde), die durch das Projekt- und Workflowmanagementsystem verarbeitet und in die Projektgraphdarstellung eingegliedert werden könnten.

Andere vorstellbare Funktionen wären die bereits während des Systemtests angesprochenen Möglichkeiten, einzelne Prozesse mehrfach (inkrementell) durchlaufen zu können oder den Abschluss einer Aktivität von der Bestätigung mehrerer Benutzer abhängig zu machen. Weitere aus dem Workflowbereich bekannte Regeln und Abhängigkeiten könnten implementiert werden, um beispielsweise bedingte Zustandsübergänge modellieren zu können (Exceptions, FollowIfNotCompleted)

Für zukünftige Arbeiten im Bereich dieser Diplomarbeit wäre daher nochmals die Frage zu diskutieren, inwieweit die Kombination eines Projekt- und Workflowmanagementsystems in einer Anwendung sinnvoll oder ob die Trennung beider Anwendungsbereiche in der Praxis besser nutzbar ist, wenn mehrere Systeme mit unterschiedlichem Schwerpunkt gegebenenfalls kooperativ agieren, indem Daten über fest vordefinierte Schnittstellen ausgetauscht werden. Dabei ist es überlegenswert, ein konzeptionell ähnliches System als reine Workflowmanagementanwendung zu entwerfen, welche sich stärker auf einen bereits bestehenden Standard wie XPDL stützt. Die Beschreibung der XML Process Definition Language im vorausgegangenen Abschnitt hat gezeigt, dass darin bereit eine Reihe an Eigenschaften (properties) definiert ist, welche direkt in eine entsprechende Ontologie überführt und unter einem semantischen Aspekt weiterverarbeitet werden können.

Bei einer Neukonzeption eines reinen semantikbewussten Workflowmanagementsystems könnte man sich im Weiteren stärker an das Workflow-Referenzmodel der WfMC halten und dadurch ein schlüssigeres System entwickeln, welches den Projektaspekt weitestgehend ausblendet.

6.4. Zukunft des Semantic Web

Als Abschluss der Diplomarbeit soll versucht werden, aus Sicht des Autors einen Ausblick auf eine mögliche Weiterentwicklung der vorgestellten Ansätze in einem zukünftigen Semantic Web zu geben.

Aus den Ergebnissen der Arbeit ist ersichtlich geworden, dass Microformats und RDFa letztendlich zwei Ziele verfolgen. Erstens sollen damit zusätzliche Informationen in bereits existierende Internetseiten eingebettet werden können, welche eine bessere maschinelle Auswertung des enthaltenen Inhalts erlauben. Zum Anderen verfolgen beide Ansätze das Ziel, dass dieser Annotationsprozess prinzipiell für jeden Endanwender durchführbar ist, indem eine Reihe an Eigenschaften durch die Verwendung von Attributen direkt vergeben werden können („tagging“). Dies kann so weit gehen, dass der Annotationsprozess für den Nutzer zunehmend transparent wird und bei einer Webentwicklung in naher Zukunft intuitiv durchgeführt wird ohne sich über den Zusatzaufwand bewusst zu sein. Eine Unterstützung dafür ist bereits heute in einigen Anwendungsprogrammen, beispielweise als Dreamweaver-Erweiterung, vorhanden.

Zusatzwissen über die zugrunde liegenden Syntaxschemas und Ontologien wäre für einen Benutzer zwar weiterhin nützlich, aber für eine Anwendung der Konzepte nicht zwingend erforderlich. Gleichzeitig muss einschränkend festgestellt werden, dass mit den heutzutage vorhandenen Konzepten das Internet kurzfristig gesehen nicht revolutioniert werden kann. Die Auszeichnung von Daten auf Webseiten mit semantischen Informationen scheint ein kritischer Schritt zu sein, allerdings werden Computer dadurch allein nicht intelligent oder sogar ihrer Umgebung bewusst werden. Algorithmen können in intelligenterer Art und Weise auf Daten operieren und mitunter bessere Ergebnisse zurückliefern. Ansonsten unterscheidet sich der Entwicklungsprozess aus Programmierersicht nicht wesentlich von bisherigen Vorgehensweisen. Der plattformübergreifende Austausch von Informationen wird in einer Art und Weise vereinfacht, wie dies durch alleinige Verwendung von DTDs oder XSDs in einigen Fällen nicht möglich wäre. Kollaborative Arbeit kann dadurch zukünftig gefördert und unterstützt werden. Momentan stellt sich diese Frage noch nicht, da die breite Verbreitung von RDFa auf verschiedensten Webseiten sicher noch auf sich warten lassen wird (mindestens bis zur offiziellen Verabschiedung der XHTML2.0-Recommendation). Bis dahin wird eine semantikbasierte Wissensverwaltung zunehmend in Softwaresystemen größerer Unternehmen ihre Verbreitung finden, wie dies heutzutage beispielsweise schon bei der NASA (xtech06.usefulinc.com/schedule/paper/147) oder am Nokia Research Center (NRC) der Fall ist.

An dieser Entwicklung werden Microformats und RDFa einen bedeutenden Anteil haben. Beide Konzepte besitzen momentan Vor- und Nachteile. Ob beide Entwicklungen langfristig Bestand haben werden und nebeneinander existieren können, vermag ich nicht zu sagen. Wenn man die gewonnenen Erkenntnisse der Diplomarbeit zusammenfasst, so könnten RDFa und Microformats als zwei Extreme der gleichen Idee gesehen werden, nämlich RDF statements in XHTML-Dokumente einbetten zu können („embedded RDF“). Der Ausdruckskraft von RDFa steht die intuitive Benutzbarkeit durch Benutzer aus dem Mainstreambereich entgegen. Kapitel 3.4. hat gezeigt, dass der Aufwand zum Entwurf einer Ontologie nicht zu unterschätzen ist und eher für abgegrenzte Softwaresysteme interessant sein könnte als für den einfachen Webseitenbetreiber. Microformats hingegen sind einfach und unkompliziert zu benutzen, vermögen jedoch nur gewisse Informationen semantisch auszeichnen zu können. Statt von zwei unvereinbaren Endpunkten zu sprechen, könnten Microformats jedoch auch als Spezialfall von RDFa aufgefasst werden, wie bereits in Abbildung 28 in Kapitel 6.1. versucht wurde zu illustrieren. So lassen sich Microformat-Beschreibungen nicht nur in RDF transformieren, sondern es existieren inzwischen Ansätze, diese mithilfe einer XSLT (hGRDDL) auch in RDFa zu überführen und so eine Verbindung zwischen beiden Konzepten herzustellen. In diesem Zusammenhang könnten Microformats als ein Spezialfall von RDFa aufgefasst werden, wobei die konkrete Anwendung in einem HTML-Dokument sich natürlich weiterhin anhand der benutzten HTML-Attribute voneinander unterscheidet.

Fest steht bisher nur, dass das Semantic Web nicht mehr nur eine Fiktion ist, sondern sich die Visionen führender Experten in naher Zukunft umsetzen lassen werden, wie der Prototyp einer semantikbasierten Projektmanagementanwendung im Rahmen dieser Diplomarbeit gezeigt hat. Eine kontinuierliche Weiterentwicklung der bestehenden Ansätze ist dabei genau so wichtig wie die Publikmachung der neuen Konzepte im Entwickler- und Endanwenderbereich um auf längere Sicht einen genauso weit verbreiteten Einsatz zu erreichen wie dies anderen Technologien in den letzten Jahren bereits gelungen ist (HTML, XML, SOAP, …) Das World Wide Web Consortium veröffentlichte dazu bereits im Jahr 2004 ein Modell des „Internets von morgen“, wie es sich von dem ursprünglichen Internet aus dem Jahr 1990 weiterentwickelt hat und welche Schichten und Konzepte es umfassen wird (Abbildung 31). Neben Web Services und XML als zugrunde liegendes Datenformat wird darin dem Semantic Web eine wesentliche Bedeutung zugesprochen.

Wann das Semantic Web kommen wird, ist dabei die falsche Fragestellung, denn es findet sich bereits an viele Stellen, wenn auch in kleinerer Form. Ob das XHTML Friend Network (XFN)– ein erstes Microformat aus dem Jahr 2003, um Links zu anderen Webseiten eine grundlegende Bedeutung geben zu können – Dublin Core oder RSS – semantische Metainformationen finden sich bereits an vielen Stellen. Ebenso wurden bereits Projekte wie Annotea (www.annotea.org) getestet, in denen Nutzer auf Webseiten eigenständig weiterführende Informationen zu den bereits vorhandenen Daten hinzufügen können. Es werden in Zukunft sicherlich eine Reihe weiterer Anwendungen mit Bezug zum Semantic Web die Aufmerksamkeit erregen, doch wird sich der Formierungsprozess des heutigen Internets zu einem „Semantic Web“, im Gegensatz zu einem immer wieder aufkommenden Hype rund um diese Thematik, sicher nur langsam vollziehen.

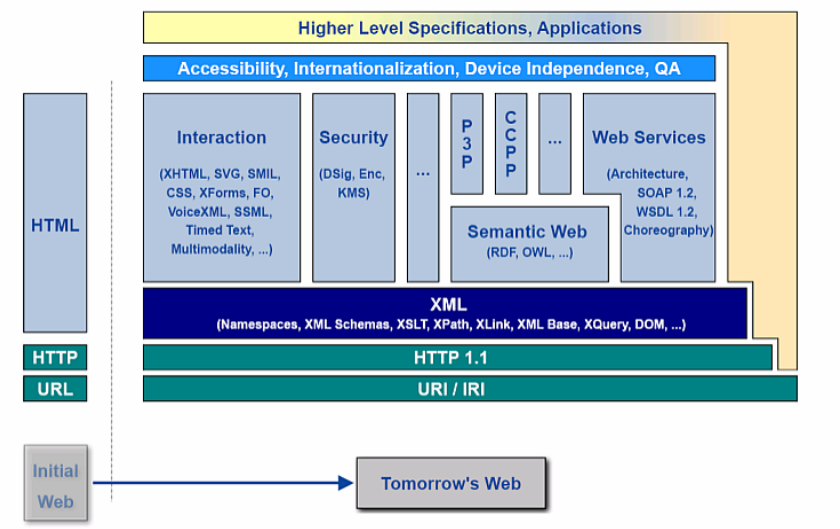


Abbildung 31: Evolution of Web Technologies, Quelle: http://www.w3.org/2004/Talks/1109-sb-gsaWebSci/slide12-0.html

Literaturverzeichnis

[Aal98] Van der Aalst, Wil: *Formalization and Verification of Event-driven Process Chains*. In: Computing Science Reports 98/01, Eindhoven University of Technology, Eindhoven, 1998.

[Bau04] Bauer, Thomas: *Kooperation von Projekt- und Workflow-Management-Systemen*. In: Informatik - Forschung und Entwicklung, Springer Berlin / Heidelberg, Vol. 19, Ausgabe 2, November 2004

[Ben01] Berner, Winfried: *Projekte – Definition und Nutzen*, URL: http://www.umsetzungsberatung.de/projekt-management/projekte.php, Abruf: 15.09.2007 12:23 Uhr

[Ber01] Berners-Lee,Tim; Hendler, James; Lassila, Ora: *The Semantic Web*. In: Scientific. American, Issue 5, May 2001

[Ber99] Berners-Lee, Tim; Fischetti, Mark: *Weaving the Web*. Harper San Francisco, October 1999

[Bod03] Bodendorf, Freimut: *Daten- und Wissensmanagement*, Springer-Verlag, Berlin, Heidelberg, New York, 2003

[Bre07] Breitman, Karin; Casanova, Marco Antonio, Truszkowski, Walter: *Semantic Web – Concepts, Technologies and Applications*, Springer-Verlag, London 2007

[Cra01] Cranefield, Stephen: *UML and the Semantic Web*, In: Proceedings of the International Semantic Web Working Symposium SWWS, Palo Alto, July 30- August 1, 2001.

[Dav93] Davenport, Thomas: *Process Innovation - Reengineering Work Through Information*. Harvard Business School, Boston, 1993

[Fer97] Ferstl, Otto; Sinz, Elmar: *Modeling of Business Systems Using the Semantic Object Model (SOM) - A Methodological Framework*. Bamberger Beiträge zur Wirtschaftsinformatik Nr. 43, Juli 1997

[Gad03] Gadatsch, Andreas: *Grundkurs Geschäftsprozessmanagement*, 3. Auflage, Wiesbaden 2003

[Gae05] Gaedke, Martin; Meinecke, Johannes; Nussbaumer, Martin: *A Modeling Approach to Federated Identity and Access Management* (Conference Paper) In: Proceedings of the Fourteenth International World Wide Web Conference (WWW), Pages 1156-1157, Chiba, Japan, 10-14 May, 2005

[Gru93] Gruber, Tom. *A translation approach to portable ontologies*. Knowledge Acquisition, 5(2):199-220, 1993, URL: http://www-ksl.stanford.edu/kst/what-is-an-ontology.html, Abruf: 09.09.2007: 18:53 Uhr

[Gua98] Guarino, Nicola: *Formal ontology and information systems*. In N. Guarino (ed.), Formal Ontology in Information Systems. Proceedings of the First International Conference (FOIS'98), June 6-8, Trento, Italy (pp. 3-15). Amsterdam

[Hab98] Habermann, Frank; Wargitsch, Christoph: *IMPACT: Workflow-Management-System als Instrument zur koordinierten Prozessverbesserung*, Universität Erlangen-Nürnberg 1998

[Ham93] Hammer, Michael; Champy, James: *Business Reengineering - Die Radikalkur für das Unternehmen*. Frankfurt/New York (Campus) 1993

[Hau02] Haun, Matthias: *Handbuch Wissensmanagement*. Heidelberg: Springer 2002

[Hen01] Hendler, James: *Agents and the Semantic Web*, preprint for the IEEE Intelligent Systems Journal, April 2001. URL: http://www.cs.umd.edu/~hendler/AgentWeb.html, Abruf: 19.10.2007 13:26 Uhr

[Hun06] Hundhausen, Richard: *Working with Microsoft Visual Studio 2005 Team System*, Microsoft Press 2006

[Ike06] Schaffert, Sebastian; Westenthaler, Rupert; Gruber, Andreas: *Ike Wiki: A User-friendly Semantic Wiki*. Salzburg Research Forschungsgesellschaft, 2006

[Joh06] John, Michael: *Glossareintrag Workflow*, Faunhofer IESE, Kaiserslautern, 2006; URL: http://vsek.org/?2368, Abruf: 05.08.06 18:46 Uhr

[Käm00] Kämpf, Rainer; Pietzsch, Marco: *Vom Taylorismus zu Prozessorientierung und Workflow-Management*, URL: http://www.ebz-beratungszentrum.de/organisation/themen/orgproz_1.htm, Abruf: 24.07.2007 15:23 Uhr

[Kin03] Kindler, Ekkart: *On the semantics of EPCs - A framework for resolving the vicious circle*. Technical Report, Reihe Informatik tr-ri-03-243, Institut für Informatik, Universität Paderborn, 2003

[Kir94] Kirn, Stefan; Unland, Rainer: Workflow *Management mit kooperativen Softwaresystemen - State of the Art und Problemabriss*, Institut für Wirtschaftsinformatik der Westfälischen Wilhelms-Universität Münster 1994

[Lac05] Lacy, W. Lee: *OWL – Representing Information Using the Web Ontology Language.* Trafford Publishing Ltd., 2005

[Möl00] Möller, Thor: *Projektmanagement in der Entwicklungszusammenarbeit,* In: Septissima, 2000, URL: http://www.con-thor.de/Projektmanagement%20in%20der%20Entwicklungszusammenarbeit.pdf, Abrufzeit: 24.07.2007 14:35 Uhr

[Now07] Nowack, Benjamin: *A Comparison of Microformats, eRDF, and RDFa,,* URL: http://bnode.org/blog/2007/02/12/comparison-of-microformats-erdf-and-rdfa, Abruf: 12.10.2007 18:43 Uhr

[Pas04] Passin, Thomas: *Explorer's Guide to the Semantic Web*, Manning Publications Co, 2004

[Pet99] Peters, Ralf: *Business Objects, Workflow und die UML - Ein Entwurf zu Vorgehensweise und Referenzmodell bei der Konstruktion von workflow-orientierten 3-Schicht Applikationen.*, In: OBJEKTspektrum 3/1999, 69-73

[PMI04] N.N.: *A Guide to the Project Management Body of Knowledge* - PMBOK Guide. Third Edition, 2004

[Pro06] Prodromou, Evan: *RDFa vs. Microformats,,* URL: http://evan.prodromou.name/RDFa_vs_microformats, Abruf: 12.10.2007 18:42 Uhr

[Rus95] Rusinkiewicz, Marek; Sheth, Amit: *Specification and Execution of Transactional Workflows*. In Modern Database Systems, W. Kim (Hrsg.). Reading MA, Addison-Wesley, 1995, pp. 592-620.

[Sch00] Schreiber, Andreas: *RDF und XML - Möglichkeiten für digitale Publikation und Archivierung*. Vortrag an der Technischen Universität Chemnitz, 2000. URL: http://archiv.tu-chemnitz.de/pub/2000/0042/data/vortrag.html, Abruf: 28.07.08 14:13 Uhr

[Sch06] Schmitz, Christoph; Hotho, Andreas; Jäschke, Robert; Stumme, Gerd: *Kollaboratives Wissensmanagemen,* In: Semantic Web - Wege zur vernetzten Wissensgesellschaft, Springer Berlin Heidelberg, 2006

[Sin05] Sinz, Elmar.: *Das Semantische Objektmodell (SOM).* Universität Bamberg, URL: http://www.unibamberg.de/en/fakultaeten/wiai/faecher/wirtschaftsinformatik/seda/leistungen/forschung/forschungsprojekte/das_semantische_objektmodell_som/, Abruf: 10.08.2007 16:21 Uhr

[Sow03] Sowa, John: *Ontology – Words of Wisdom*, URL: http://www.jfsowa.com/ontology/index.htm, Abruf: 09.09.2007 18:02 Uhr

[Sur06] Sure, York; Tempich, Christoph: *Wissensvernetzung in Organisationen*. In: Tassilo Pellegrini and Andreas Blumauer , Semantic Web - Wege zur vernetzten Wissensgesellschaft, chapter 3.e.. Springer, Heidelberg, X.media.press , 2006

[Tur50] Turing, Alan: *Computing Machinery and Intelligence*. Mind, Oct 150, 433-460

[Ver07] N.N. : *The VeriSign Domain Report*. The Domain Name Industry Brief, Volume 4, Issue 1, March 2007

[Wah06] Wahlster, Wolfgang: *Semantische Wende – Informatik für den Menschen*. Festvortrag im Rahmen der Abschlussveranstaltung des Informatikjahres des BMBF, Berlin, 18.12.2006, URL: http://www.informatikjahr.de/fileadmin/content/documents/Reden/Festrede_Wahlster_Abschluss_Informatikjahr.PDF, Abruf: 09.10.2007 16:03 Uhr

[Wen06] Wenning, Rigo: *Was wir von Google lernen können – Die ungebremste Kreativität der Web-Suche*. 26.04.2006, URL: http://www.eear.eu/fileadmin/wenning/, Abruf: 26.07.07 11:53 Uhr

[Zoll06] Zoll, Oliver: *Der Business Rules Approach (BRA) in der Praxis - Regelbasierte Prozessmodellierung am Beispiel eines betrieblichen Pensionsplans*. Fachhochschule für Oekonomie & Management, Frankfurt am Main, 2006

Index

A

AJAX 62, 103, 113, 123
Aktivität 40
Anwendungsszenarien 50, 119

B

BPMN 49, 70, 116

D

Dokument 23
Dublin Core 80, 89, 96

E

eRDF 90
Ereignisgesteuerte Prozessketten 45
Evaluierung 54

F

Flussdiagramm 46
FOAF 88

G

GRDDL 91, 105, 111, 113
Groupware 42

H

hCalendar 95
hGRDDL 94
HTML 22, 104, 108, 113, 127

I

idFS 109, 113, 114, 125
Internet 26

K

Kontext 76

M

Metadaten 24, 89, 125
Microformats 91, 92, 108, 123, 130, 133

N

N3 84
N-Tripels 83

O

Ontologie 80, 84, 96, 97
OWL 28, 85, 88

P

Petri-Netze 46
Phase 36
Profil 105
Projekt 35
Projektmanagement 32, 33, 40, 56, 71
Prozess 37
Prozessmanagement 41

R

RDF 70, 82, 88, 113
RDFa 90, 92, 105, 108, 111, 114, 123, 130, 133
RDFS 84, 88, 96, 102

S

Semantic Web 27, 132
Semantic Wiki 32
Semantik 75, 76
Semantisches Schema 79
SGML 22
SHOE 88
Syntactic Web 23
Systemarchitektur 107, 123

T

Topic Maps 86
Turtle..... 84

U

UML..... 48

V

vCard..... 95

W

W3C.....28, 82, 85
Wiki 62
Wissen 77
Wissensmanagement.....43
Workflow.....38
Workflow Management Coalition 110
Workflow-Engine40, 63, 72, 110, 114, 117, 124, 129
Workflowmanagement 32, 42, 63, 71

X

XML 22, 25, 76, 81, 127
XPDL.....49, 63, 66, 98, 110, 128
XSD 25, 133
XSLT 23

A n h a n g

A. Anhang

A.1. Getestete Projektmanagementsysteme

Produktname	PHProjekt
Generelle Informationen	
Hersteller	Mayflower GmbH, Deutschland
Version	5.2
Webseite	http://www.phprojekt.com/
Entwicklungsbeginn	2000
Letzte Aktualisierung	20.12.2006
Managementsystem-Kategorie	Projektmanagement, Groupware
Plattform	Web-basiert
Programmiersprache	PHP4 in Verbindung mit MySQL, PostgresQL, Oracle, Informix, Interbase, MS-SQL, SQLite
Datenmodell	Proprietär
Addons / Plugins vorhanden	Ja, zahlreiche zur Integration anderer Systemkomponenten wie Wikis, WebDav-Funktionen oder spezieller Ressourcenübersichten.
Sprachen	Deutsch, viele weitere Sprachpakete verfügbar
Lizenz	GPL
System	
Funktionsgruppen	Kalender, Aufgaben, Helpdesk, Umfragen, Kontakte, Forum, Dateien, Projekte, Notizen, Mail, Chat, Zeitkarte, Lesezeichen für relevante URLs
Navigationsstruktur / Bedienkonzept	Textlink-basierte Navigation links, Übersichtsseite mit Kurzübersicht zu aktuellen Aufgaben, Projekten, Terminen u.ä.
Layout	4 verschiedene Basis-Skins auswählbar, weitere Skins installierbar, Trennung von Navigations- und Inhaltsbereich, ansonsten sehr textlastig
Graphische Modellierwerkzeuge	Keine
Modulübergreifende Datenauswertung	Stark eingeschränkt, nur in Form von Nutzernamen, Projektbezeichnungen und Kontaktdaten findbar
Intuitive Bedienbarkeit	Strikte Domaintrennung, Formulareingabemasken mit detaillierten Abfragen, spontane Modellierung schwierig
Nutzerverwaltung	Nutzernamen und Nutzergruppen (Normaler Nutzer, Benutzer mit Chef-Rechten, Administrator) für Authentifizierung, darüber hinaus weitere Rollen definierbar, unabhängig davon Projektrollen und Kontaktverteiler definierbar
Aufgabenverwaltung	Einzelne Aufgaben mit Termin, Priorität und Beschreibung definierbar und einem Projekt zuordbar, Aufgaben aber nicht hierarchisch weiter unterteilbar oder zueinander zeitlich in Beziehung setzbar, kein Template- oder Versionssystem zur Wiederverwendung, Statusauswahl wartend, offen, angenommen, abgelehnt, beendet
Strukturierung von Aktivitäten	Nicht strukturierbar, einfache Aufgabenlisten pro Projekt
Dokumentenverwaltung	Dateiupload getrennt nach Projekt unterstützt, in Verzeichnissen organisierbar und mit Passwort vor unberechtigtem Zugriff schützbar
Zeit- und Forschrittsauswertungen	Nur anhand von Prioritäten oder Statusmarkierungen, keine Gantt-Diagramme oder andere graphischen Übersichten
Systembenachrichtigungsfunktionen	E-Mail-Benachrichtigungsfunktion bei Zuteilung neuer Aufgaben
Kommunikationsfunktionen	E-Mail zur Kontaktaufnahme
Weitere Groupware-Funktionen	Kalender, Chat, Forum
Anwendungsszenarien	
Szenario 1: Spaghetti kochen	Nur als einzelne Aufgabenbeschreibung in Textform definierbar, entweder projektunabhängig oder einem Pseudo-Projekt zu einem bestimmten Datum zuordbar
Szenario 2: Abendessen mit Freunden	Projekt kann hierarchisch in mehrere Teilprojekte unterteilt werden, Aufgabenzuteilung zu verschiedenen Nutzern und Zeitabhängigkeit aber für sinnvolle Modellierung nicht ausreichend
Szenario 3: Diplomarbeit schreiben	Mangelnde modulübergreifende Datenaufbereitung stellt sich bei komplexeren Projekten als erschwerend heraus. Filterfunktionen ermöglichen zwar, nur bestimmte Projektteile anzuzeigen, aber die Gesamtstrukturierung ist nur durch die Definition weiterer Subprojekte möglich, wodurch das Projektmanagementsystem mehr zu einer Art Tabel-

	lenverarbeitung wird
Szenario 4: Applikationsentwurf	Für Bugtracking und Delegation einzelner Aufgaben gut geeignet, ansonsten ähnliche Probleme wie in Szenario 3, obwohl derartige geschäftsnahe Prozesse durch ein gut integriertes Ressourcen- und Zeitmanagement besser unterstützt werden
Szenario 5: Workshop organisieren	Aufgrund mangelnder Intuitivität / zu komplexer Formularmasken nur schwer für große Gruppenprojekte benutzbar, Planung muss vorher von Hand durchgeführt werden, Anwendung kann nur bei praktischer Umsetzung zur kollaborativen Umsetzung und Überwachung genutzt werden
Gesamteindruck	
Hilfreiche Funktionen	Arbeitszeitkarte und Projektstoppuhr, Lesezeichen für projektbezogene URLs
Fehlende Funktionen	Versionierung, Vernetzung von Projekten, Aktivitäten und Akteuren, grafische Übersichten
Bewertung	Insgesamt eine Groupware-Lösung, welche einfache Projekte in kleinen Teams über einen längeren Zeitraum gut unterstützt, bei komplexeren Projekten nicht praktikabel, da Projekte, Aufgaben, Nutzer, Notizen und Dateien so gut wie vollständig unabhängig voneinander verwaltet werden.

Produktname	Double Choco Latte
Generelle Informationen	
Hersteller	Michael Dean / Free Software Foundation (Source-forge.net)
Version	0.9.5
Webseite	http://dcl.sourceforge.net/
Entwicklungsbeginn	18.11.1998
Letzte Aktualisierung	07.01.2007
Managementsystem-Kategorie	Projektmanagement, Bugtracking
Plattform	Web-Basiert
Programmiersprache	PHP in Verbindung mit MySQL, PostgreSQL, Sybase oder MSSQL
Datenmodell	Proprietär
Addons / Plugins vorhanden	Nein
Sprachen	Englisch
Lizenz	GPL
System	
Funktionsgruppen	Work Orders, Projects, Tickets, Manage, Help
Navigationsstruktur / Bedienkonzept	Wichtigste Menupunkte aus Kategorien getrennt in horizontaler Navigationsleiste je nach Accountpermissions, Übersichtsseite mit den jeweils fünf neuesten Aufgaben, vergebenen Aufgaben und Projekten, sowie weitere inhaltsabhängige Kurznavigation am linken Bildrand
Layout	Klar strukturiert und übersichtlich
Intuitive Bedienbarkeit	Neues Projektanlegen durch Spezifikation eines Titels, eines Hauptverantwortlichen und einer Deadline sowie der möglichen Zuordnung eines übergeordneten Projekts. Navigation wirkt aufgeräumt, alle benötigten Funktionen finden sich schnell und einfach
Graphische Modellierwerkzeuge	Keine
Modulübergreifende Datenauswertung	Ja, Work-Order-Tätigkeiten erscheinen in Projektkonfiguration und können von dort in Abhängigkeit von den Zugriffsrechten auch direkt editiert werden
Nutzerverwaltung	Feste Rollen: guest, support, programmer, manager
Aufgabenverwaltung	Nur zwei Aufgabentypen: Bug oder Enhancement, der eine Deadline und eine Beschreibung zugeordnet wird
Strukturierung von Aktivitäten	Work Orders können einzelne Aufgaben zugeordnet werden, wodurch eine flache Struktur entsteht, komplexe Workflows darin aber nicht abbildbar sind. Nur durch Projekthierarchie realisierbar
Dokumentenverwaltung	Zu einem Projekt können Dateien hochgeladen und zugeordnet werden
Zeit- und Forschrittsauswertungen	Nur anhand von Prioritäten oder Statusmarkierungen, keine Gantt-Diagramme, aber z.B. Aktivitätsdiagramme über offene und abgeschlossene Work Orders
Systembenachrichtigungsfunktionen	Email-Benachrichtigungsfunktionen vorhanden
Kommunikationsfunktionen	Organisationen/Kontakverwaltung vorhanden
Weitere Groupware-Funktionen	Wiki zur Arbeit an Projekten, Forum/FAQ-Bereich
Anwendungsszenarien	
Szenario 1: Spaghetti kochen	Als Work Order abbildbar, wenn auch Systemkategorien irreführend sind (Bug/Enhancement)
Szenario 2: Abendessen mit Freunden	Als Projekt abbildbar, welches aus einzelnen Work Orders besteht
Szenario 3: Diplomarbeit schreiben	Als Projekt eingeschränkt abbildbar, welches aus mehreren Subprojekten besteht, welche widerum aus einzelnen Aufgaben bestehen
Szenario 4: Applikationsentwurf	Als Projekt eingeschränkt abbildbar, welches aus mehreren Subprojekten besteht, welche widerum aus einzelnen Aufgaben bestehen, auch wenn Systemfokus auf einer Arbeit an einer bereits bestehenden Applikation liegt
Szenario 5: Workshop organisieren	Ebenfalls als Projekt aus mehreren Subprojekten abbildbar, aber Rollenverteilung geht fast vollständig verloren. Zur Verwaltung in der Praxis würde ein anderes Managementsystem benutzt werden
Gesamteindruck	
Hilfreiche Funktionen	Metriken und andere Statistikauswertungs- und Aggregatfunktionen (offene Work Orders, …)
Fehlende Funktionen	Generalistischere Eingabemasken, welche auch IT-unabhängige Projekte managen lassen, Versionierung von Dateien und Beschreibungen, grafische Darstellung der zeitlichen Abfolge von Work Items und Task Lists
Bewertung	Für IT-Projekte und im Supportbereich ein gutes System,

	zum Projektmanagement im Geschäftsalltag oder für IT-fremde Einsatzgebiete ungeeignet

Produktname	dotproject
Generelle Informationen	
Hersteller	Unabhängiges Entwicklungsteam (Adam Donnison and Gryphon, Gregor Erhardt, Ivan Peevski et al.)
Version	2.1.0 rc2
Webseite	http://www.dotproject.net/
Entwicklungsbeginn	2000
Letzte Aktualisierung	21.05.2007
Managementsystem-Kategorie	Projektmanagement
Plattform	Webbasiert
Programmiersprache	PHP in Verbindung mit MySQL oder ADOdb
Datenmodell	Proprietär
Addons / Plugins vorhanden	Ja, als dotmods bezeichetes unabhängiges Projekt, welches u,A. ein Backup-Modul, Heldesksupport und einen Projektdesigner beinhaltet
Sprachen	Insgesamt 73 verschiedene Sprachpakete vorhanden
Lizenz	GPL
System	
Funktionsgruppen	Companies, Projects, Tasks, Calendar, Files, Contact, Forums, Tickets, Admin
Navigationsstruktur / Bedienkonzept	Horizontale Navigation zwischen einzelnen Funktionsgruppen, Übersichtsseite mit heutigen Aufgaben und Ereignissen
Layout	Prinzipiell nur ein Skin verfügbar mit schlichtem, Windows-ähnlichem Design, Projektstatus wird farblich hervorgehoben. Außerdem können Projekten eigene Erkennungsfarben zugeordnet werden, wodurch in Übersichtslisten jedoch an manchen Stellen gerade dadurch die Übersichlichkeit verloren geht
Intuitive Bedienbarkeit	Weitestgehend ja, klare Abgrenzung der Module. Einige Funktionen sind aber auch eine Navigationsebene tiefer angeordnet und dadurch etwas versteckt (bspw. Nutzer zu einem Projekt/einer Aufgabe zuordnen)
Graphische Modellierwerkzeuge	Keine, Projekte können aber farblich verwaltet werden und Aufgaben in einer Listenansicht umsortiert werden
Modulübergreifende Datenauswertung	Ja, Aufgaben und Kontakte werden in Projektmodul automatisch integriert, Kalender übernimmt Projektaufgaben und Termine
Nutzerverwaltung	Frei konfigurierbare Nutzerrollen wie admin, normal, anonymous, guest user
Aufgabenverwaltung	Zu Projekt zugeordnete Aufgaben können hierarchisch weiter untergliedert werden, können umsortiert werden, als Aufgabe mit Meilensteincharakter ausgezeichnet werden und in ihrer Abfolge in einem Gantt-Diagramm dargestellt werden
Strukturierung von Aktivitäten	Ja
Dokumentenverwaltung	Eigenes Dokumentenmodul, über welches zu Projekten und Aufgaben Dateien zugeordnet werden können. Diese sind in einer Ordnerstruktur organisierbar und können mit Attributen versehen werden (Version, Kategorie, Beschreibung)
Zeit- und Forschrittsauswertungen	Ja, Report- und Diagrammfunktionen
Systembenachrichtigungsfunktionen	Ja
Kommunikationsfunktionen	Kontaktablage/Adressbuch
Weitere Groupware-Funktionen	Forum
Anwendungsszenarien	
Szenario 1: Spaghetti kochen	Im Rahmen des Projekts von Szenario 2 als einfache Aufgabe mit mehreren Unteraufgaben abbildbar, auch wenn die Datumsauswahlfunktion für jeden Schritt nicht sinnvoll erscheint
Szenario 2: Abendessen mit Freunden	Als Projekt mit mehreren Aufgaben abbildbar
Szenario 3: Diplomarbeit schreiben	Ebenfalls als Projekt mit mehreren Aufgaben modellierbar. Es fehlt eine Möglichkeit, ein Projekt aus mehreren Subprojekten (Phasen) aufzubauen, was in diesem Fall über einzelne Aufgaben zu geschehen hat, wobei abschließende Aufgaben als Meilenstein gekennzeichnet werden
Szenario 4: Applikationsentwurf	Siehe Szenario 3. IT-Projekte sind mit dotproject gut realisierbar, da spezielle Eingabemasken für Termine, Arbeitsstunden, Projektbudget und Entwicklungs-URLs bereitstehen
Szenario 5: Workshop organisieren	Ist mit dem Projektmanagementsystem auch als Projekt umsetzbar, wenn auch komplex. Die Zuordnung von Unternehmen und Verantwortlichen ist hilfreich, eine Unterstüt-

	zung durch das System in einzelne Teilprojekte wäre wünschenswert
Gesamteindruck	
Hilfreiche Funktionen	Definition von Events, Meilenstein-Markierung und Aufgabentagging, Auswertung als Gantt-Diagramm, Dateiverwaltung mit Versionsinformationen
Fehlende Funktionen	Strukturierung von Projekten in Teilprojekte
Bewertung	Dotproject ist ein webbasiertes Projektmanagementsystem, welches kaum eine Funktion vermissen lässt und professionell eingesetzt werden kann, trotz dass es kostenlos ist

Produktname	WebCollab
Generelle Informationen	
Hersteller	Unabhängiges Entwicklungsteam (Andrew Simpson et al.)
Version	2.11
Webseite	http://webcollab.sourceforge.net/
Entwicklungsbeginn	2002
Letzte Aktualisierung	04.02.2007
Managementsystem-Kategorie	Projektmanagement
Plattform	Webbasiert
Programmiersprache	PHP mit MySQL oder PostgreSQL
Datenmodell	Proprietär
Addons / Plugins vorhanden	Nein
Sprachen	Ja, über 25 verschiedene Sprachpakete dabei
Lizenz	GPL
System	
Funktionsgruppen	Projektübersicht, ToDo-Listen, Kalender, Forum, Archiv, Nutzer, Nutzergruppen
Navigationsstruktur / Bedienkonzept	Zentrale Navigation am linken Bildschirmrand. Gesamtübersichtsseite über alle laufenden Projekte und deren Status, weitere Projektübersichtsseite, separater Menubereich zur Verwaltung von Aufgaben und Projekten sowie Nutzer(-gruppen)
Layout	Schlicht aber funktionell, auf Dateisystemebene anpassbar
Intuitive Bedienbarkeit	Ja
Graphische Modellierwerkzeuge	Nein, nur graphische Fortschrittsbalken und –grafiken über aktuellen Projektstatus
Modulübergreifende Datenauswertung	Ja, Projekt-, Aufgaben- und Nutzerdaten werden automatisch verarbeitet und aufbereitet
Nutzerverwaltung	Einfache Nutzerverwaltung, Anmeldung mittels Benutzername und Passwort, Gruppierung durch frei definierbare Nutzergruppen, daneben Kontaktlistenfunktion
Aufgabenverwaltung	Beliebig viele Aufgaben einem Projekt zuordbar, Aufgaben hierachisch untergliederbar, Übersicht an aktuell zu bearbeitenden Aufgaben in spezieller ToDo-Liste
Strukturierung von Aktivitäten	Hierarchische Gliederung möglich, allerdings können Aktivitäten nur durch Start- und Enddatum in eine zeitliche Reihenfolge gesetzt werden, die allerdings vom System nicht weiter ausgewertet oder dargestellt wird, bestehende Aufgaben sind darüber hinaus klonbar und können so wieder verwendet werden
Dokumentenverwaltung	Sowohl Projekten als auch Aufgaben können Dateien zugeordnet werden
Zeit- und Forschrittsauswertungen	Grafische Darstellung relativ zu den Deadline-Angaben, Gantt-Diagramme oder Gesamtdarstellung der zeitlichen Abfolge fehlen, dafür Eventmanagement vorhanden, wann von wem etwas im System modifiziert wurde
Systembenachrichtigungsfunktionen	Emailbenachrichtigungsfunktionen vorhanden, optional für jede Projektmodifikation (de)aktivierbar
Kommunikationsfunktionen	Für jedes Projekt separates Forum vorhanden
Weitere Groupware-Funktionen	Kontaktlisten, Kalender, Archiv
Anwendungsszenarien	
Szenario 1: Spaghetti kochen	Sollte in Projekt eingebettet sein, kann dann aber als Task gut in Subtasks strukturiert werden; nur eine Deadline ist (wie bei Projektmanagementsystemen üblich) nötig anzugeben
Szenario 2: Abendessen mit Freunden	Gut als Projekt abbildbar
Szenario 3: Diplomarbeit schreiben	Projekt gut abbildbar und in Subprojekte („Phasen“) untergliederbar, unterschiedliche Rollenverteilung und Benachrichtigung bei Events durch Nutzergruppenmodell abbildbar
Szenario 4: Applikationsentwurf	Analog Szenario 3 sind auch Software Engineering Aufgaben gut abbildbar
Szenario 5: Workshop organisieren	Durch die weitreichenden Strukturierungsmöglichkeiten ist auch ein komplexes Projekt wie in diesem Szenario bis zur untersten Ebene modellierbar und überschaubar. Eine Gesamtübersicht wird durch eine Baumnavigation geboten. Die projektbezogene Forennavigation erleichtert die Kommunikation in der Umsetzungsphase
Gesamteindruck	
Hilfreiche Funktionen	Nutzergruppensystem, Untergliederung von Aufgaben in Subaufgaben, Projektbezogenes Forensystem, Fortschrittsanzeige in Bezug auf Deadline, Klonen von Projek-

	ten / Aufgaben
Fehlende Funktionen	Zeitlicher Bezug zwischen einzelnen Aufgaben, graphische Auswertungswerkzeuge, Versionierung, Individuelle Benachrichtigungsfunktionen bei Statusänderungen; außerem fehlen jegliche geschäftsbezogene Statistikfunktionen zur Ressourcenauswertung
Bewertung	Ein einfaches, aber sehr übersichtliches und effektives System zur Umsetzung einer Reihe von Aufgaben

Produktname	PHPCollab
Generelle Informationen	
Hersteller	Unabhängiges Entwicklerteam
Version	2.5 rc3
Webseite	http://phpcollab.sourceforge.net/
Entwicklungsbeginn	12.02.2002
Letzte Aktualisierung	03.06.2005
Managementsystem-Kategorie	Projektmanagement
Plattform	Webbasiert
Programmiersprache	PHP in Verbindung mit MySQL, PostgreSQL oder MSSQL
Datenmodell	Proprietär
Addons / Plugins vorhanden	Nein
Sprachen	Englisch, Französisch, Italienisch
Lizenz	GPL
System	
Funktionsgruppen	Projekte, Kunden, Berichte, Kalender, Lesezeichen
Navigationsstruktur / Bedienkonzept	Horizontale Navigationsleiste, zentrale Übersichtsseite mit Gruppierung aller relevanten Daten, von der aus alle Informationsseiten erreicht werden können
Layout	Einfaches funktionelles Layout, welches auf Dateiebene an das eigene Corporate Design anpassbar ist
Intuitive Bedienbarkeit	Intuitiv bedienbar mithilfe von Icons und Beschreibungen
Graphische Modellierwerkzeuge	Keine
Modulübergreifende Datenauswertung	Ja, Projekt- und Aufgabendaten werden auch in anderen Modulen (z.B. Kalender) benutzt
Nutzerverwaltung	Benutzerverwaltung über Administrationsschnittstelle, Anmeldung mittels Benutzername und Passwort, Gruppierung in Organisationen, feste Nutzerrollen, Zuordnung von Nutzern zu Projekten, Delegation von Aufgaben
Aufgabenverwaltung	Projekten können einzelne Aufgaben zugeordnet werden, die an einen Mitarbeiter weiterdelegiert werden und durch einen Arbeitsfortschritt charakterisiert werden
Strukturierung von Aktivitäten	Aufgaben sind in Subaufgaben weiter unterteilbar
Dokumentenverwaltung	Professionelle Dokumentverwaltung mit Versionsmanagement, alternativen Dokumenten und Freigabefunktionen
Zeit- und Forschrittsauswertungen	Fortschrittsauswertung deadline-basiert, Fortschrittsfeststellung weitestgehend manuell (prozentueller Fortschritt muss durch Verantwortlichen geschätzt und regelmäßig eingetragen werden)
Systembenachrichtigungsfunktionen	Emailbenachrichtigungsfunktion über zugeteilte Aufgaben und aktualisierte Projekte und Projektstatus
Kommunikationsfunktionen	Keine weiteren
Weitere Groupware-Funktionen	Kalender, Diskussionen
Anwendungsszenarien	
Szenario 1: Spaghetti kochen	In Projekt eingebettet als Aufgabe mit einzelnen Unteraufgaben abbildbar; einige Felder der Formularmasken sind semantisch stark am Software-Engineering orientiert, stören bei der Modellierung allgemeinerer Prozesse aber nicht
Szenario 2: Abendessen mit Freunden	Als Projekt mit einzelnen Aufgaben gut modellierbar, in einzelne Phasen unterteilbar
Szenario 3: Diplomarbeit schreiben	Als Projekt mit mehreren Phasen, Projektteilnehmern und Aufgaben beschreibbar. Dokumentmanagement mit hilfreicher Unterstützung
Szenario 4: Applikationsentwurf	Von Grund auf unterstützt durch vordefinierte Projektphasen und späteres Bugtracking und Delegation von Aufgaben
Szenario 5: Workshop organisieren	Auch mit PHPCollab umsetzbar, wenn auch das Projektmanagement an einigen Stellen für eine andere Domain konzipiert erscheint
Gesamteindruck	
Hilfreiche Funktionen	Unterstützt explizit die Modellierung von Projektphasen, ausgefeiltes Dokumentmanagement, Delegation von Aufgaben, automatisierte Berichterstellung, öffentlicher Bereich / Kundenseiten
Fehlende Funktionen	Graphische Darstellung der einzelnen Projektteile und Zuständigkeiten, insgesamt etwas zu textlastig
Bewertung	PHPCollab erscheint trotz der langen Entwicklungspause als sehr ausgereiftes, praxistaugliches System, welches insbesondere im Bereich der Anwendungsentwicklung, Darstellung von Geschäftsprozessen und Dokumentenverwaltung Stärken hat.

Produktname	Basecamp
Generelle Informationen	
Hersteller	37signals
Version	n/a
Webseite	http://www.basecamphq.com/
Entwicklungsbeginn	1999
Letzte Aktualisierung	2007
Managementsystem-Kategorie	Projektmanagement, Groupwar
Plattform	Webbasiert
Programmiersprache	Ruby on Rails
Datenmodell	Proprietär
Addons / Plugins vorhanden	n/a
Sprachen	Englisch
Lizenz	Commercial, free account available
System	
Funktionsgruppen	Project, Messages, To-Do, Milestones, Whiteboard, Chat; TimeTracking und FileStorage stehen nur in den kostenpflichtigen Accounts zur Verfügung
Navigationsstruktur / Bedienkonzept	Horizontale Navigation mit Registerreitern, wichtigste Funktionen per Ajax in räumlicher Nähe zu bearbeitendem Objekt
Layout	Layoutoutgrundgerüst vorgegeben, Farben und Logos an Corporate Design anpassbar
Intuitive Bedienbarkeit	Einfache Bedienung
Graphische Modellierwerkzeuge	ja, Aufgaben können nach Erstellung per Drag & Drop in eine beliebige Reihenfolge gebracht werden
Modulübergreifende Datenauswertung	Nein, nur Darstellung aller Eingaben auf Übersichtsseite
Nutzerverwaltung	An Projekt beteiligte Nutzer können eingeladen werden, die sich dann bspw. per OpenID authentifizieren. Außer einem Administrator haben alle Nutzer die gleichen Rechte (Collaboration)
Aufgabenverwaltung	In Form von ToDo-Listen, Aufgaben werden innerhalb eines Projektes in einzelnen Kategorien gruppiert und können an einzelne Personen delegiert werden. Eine Einschränkung des Umsetzungszeitraumes gibt es nicht, die Reihenfolge und Gruppierung wird von Hand festgelegt. Nach dem KISS-Prinzip wird lediglich der Status geändert: bearbeitet oder nicht; daneben existiert unabhängig von den ToDo-Listen die Möglichkeit, Milestones zu definieren, welche in einem Kalender eine zeitliche Aktivität kennzeichnen
Strukturierung von Aktivitäten	Nein, nur Gruppierung unter dem Titel einer bestimmten ToDo-Liste
Dokumentenverwaltung	Nur über Nachrichten-Modul, wo beliebige Dokumente angehangen werden können
Zeit- und Forschrittsauswertungen	Nur anhand des Datums der eingetragenen Milestones
Systembenachrichtigungsfunktionen	Ja
Kommunikationsfunktionen	Via Email über Liste der am Projekt beteiligten Personen
Weitere Groupware-Funktionen	Nachrichtenmodul (Art Forum), Whiteboard (Art Wiki), Chat
Anwendungsszenarien	
Szenario 1: Spaghetti kochen	Als eigene ToDo-Liste mit Anweisungen beschreibbar
Szenario 2: Abendessen mit Freunden	Termine als Meilensteine beschreibbar, Tätigekeiten ansonsten per ToDo-Liste
Szenario 3: Diplomarbeit schreiben	Obwohl ein Projekt unter Basecamp ein sehr loser Begriff is und eher eine Gruppierung kennzeichnet als ein Gebilde, dessen Zuordnungen und Constraints ständig geprüft werden, können wichtige Aufgaben von Diplomand und Betreuer mittels Basecamp erfasst werden, eine kollaborative Arbeit ist darüber hinaus mit dem Wiki-ähnlichen Whiteboard möglich
Szenario 4: Applikationsentwurf	Weniger geeignet, da es keine feste Rollenzuordnung und Aufgabentrennung in diesem Kontext gibt und keine Ressourcenverwaltung im Original-Basecamp-System existiert
Szenario 5: Workshop organisieren	Ähnlich Szenario 3: Obwohl typische Projektmanagement-Funktionen und Abhängigkeiten zwischen Aktivitäten fehlen, ist eine kollaborative Arbeit möglich, da schnell und einfach erledigte und zu erledigende Aufgaben eingetragen werden und Ideen gesammelt werden können
Gesamteindruck	
Hilfreiche Funktionen	Template-System für ToDo-Listen
Fehlende Funktionen	Abhängigkeiten zwischen den einzelnen Modulen, sodass eine Beziehung zwischen Meilensteinen und Aufgaben

	hergestellt werden kann; Ressourcenverwaltung
Bewertung	Basecamp ist ein kollaboratives System, welches den Web 2.0 – Gedanken grundlegend verfolgt. Sofern ein Projekt mit mehreren Teammitgliedern schnell und unkompliziert geplant und der Status einzelner Aufgaben überwacht werden muss, eignet sich Basecamp sehr gut

A.2. Getestete Workflowmanagementsysteme

Produktname	JaWE Java XPDL Editor
Generelle Informationen	
Hersteller	Together Teamlösungen EDV Dienstleistungen GmbH, Wien
Version	2.2-1
Webseite	http://www.enhydra.org/workflow/jawe/
Entwicklungsbeginn	1999
Letzte Aktualisierung	2007
Managementsystem-Kategorie	Workflow-Management
Plattform	Webbasiert (Java WebStart)
Programmiersprache	Java
Datenmodell	XPDL
Addons / Plugins vorhanden	Ja
Sprachen	Englisch, Deutsch, Französisch, Serbisch, Spanisch
Lizenz	LGPL
System	
Funktionsgruppen	Trennung zwischen Workflow Editor und Workflow Engine, welche eine XPDL-Beschreibung zur Ausführung bringt
Navigationsstruktur / Bedienkonzept	Windows typische GUI mit Menuleiste, Symbolleiste, Baumnavigation, Graphvorschau und Editorbereich, Workflow participants, Applikationen und Variablen werden über entsprechende Eigenschaftenfenster angelegt und können anschließend den Elementen im Workflowgraph zugeordnet werden
Layout	Windows-typisches Standardlayout
Intuitive Bedienbarkeit	Ja, abgesehen vor Vielzahl an Eigenschaften des Workflows, die optional gesetzt oder verändert werden können
Graphische Modellierwerkzeuge	Ja, XPDL-Editor
Modulübergreifende Datenauswertung	n/a, Workflow wird im XPDL-Format an Workflow-Engine übergeben, die diesen entsprechend der Beschreibung durchführt und Tests entsprechend den in dem XPDL – Dokument definierten Bedingungen durchführt
Nutzerverwaltung	Participants können beliebig erstellt und deren Aufgaben in Swimlanes verwaltet werden
Aufgabenverwaltung	Unterscheidung zwischen nicht-implementierten Aufgaben und Tool-basierten Aufgaben sowie Route acitivities zur Synchronisation
Strukturierung von Aktivitäten	Ja, Subprozesse sind möglich, ebenso Blockaktivitäten
Dokumentenverwaltung	n/a
Zeit- und Forschrittsauswertungen	n/a
Systembenachrichtigungsfunktionen	n/a
Kommunikationsfunktionen	n/a
Weitere Groupware-Funktionen	n/a
Anwendungsszenarien	
Szenario 1: Spaghetti kochen	Einfach als Workflow mit einem Participant modellierbar
Szenario 2: Abendessen mit Freunden	Als Nutzerrollen sind der Gastgeber und die Gäste möglich, die Aktivitäten können weiterhin zu activity sets zusammengefasst werden bzw wiederum einen eigenen Workflow darstellen
Szenario 3: Diplomarbeit schreiben	Die Anzahl der participants erhöht sich (Diplomand, Betreuer, Mitarbeiter, Verwaltung), für welche separate Swimlanes benutzt werden können. Die einzelnen Arbeitsschritte können durch Aktivitäten gut als Workflow abgebildet werden, Übergänge sind durch Bedingungen einschränkbar
Szenario 4: Applikationsentwurf	Rollenverteilung wird im XPDL-Editor entsprechend nach Projektmanager, Designer, Programmierer und ähnliche festgelegt, eine hierarchische Aufteilung des Gesamtprozesses ist möglich, auch wenn der Projektkontext verloren geht; dennoch stehen zusätzliche Attribute für Aktivitäten wie Priorität, Deadline oder eine textuelle Beschreibung bereit
Szenario 5: Workshop organisieren	Der XPDL-Editor erleichtert die Modellierung, auch wenn die Arbeitsoberfläche bei der Vielzahl an participants und Abhängigkeiten an ihre Grenzen stößt. Je nach Detailierungsgrad des Anwendungsszenarios sind alle Aktivitäten als Workflow abbildbar. Eine Synchronisation ist durch Route acitivities möglich
Gesamteindruck	
Hilfreiche Funktionen	Einfache Definition von Rollen, participants, Workflow graph, Übergängen und Bedingungen; Unterstützung der

	XPDL Spezifikation einschließlich Split/Join/Subprozesse, textuelle Beschreibung von Aktivitäten durch Priorität, Deadline, Text und Exceptionbehandlung
Fehlende Funktionen	Für einen reinen Workflow-Editor/Engine keine
Bewertung	Die Enhydra Community bietet einen freien XPDL Editor, mit dem Workflowbeschreibungen schnell und auf grafischem Wege erstellt und ausgeführt werden können. Nach einer ersten Erprobung fehlen keine wichtigen Funktionen aus der XPDL-Spezifikation, sodass der Editor auch für komplexere Anwendungen sinnvoll nutzbar erscheint

Produktname	Bonita
Generelle Informationen	
Hersteller	Bull R&D
Version	3.0
Webseite	http://wiki.bonita.objectweb.org/
Entwicklungsbeginn	14.02.2003
Letzte Aktualisierung	02.02.2007
Managementsystem-Kategorie	Workflow-Management
Plattform	Webbasiert
Programmiersprache	Java
Datenmodell	XPDL
Addons / Plugins vorhanden	Nein
Sprachen	Englisch
Lizenz	LGPL
System	
Funktionsgruppen	Trennung der Workflow Engine in ein Menu für Administration, Wokflow-Design, Verwaltungsfunktionen und Nutzeraufgaben, wo Workflows gruppiert nach ToStart, Running, Todo und Done abgelegt weden
Navigationsstruktur / Bedienkonzept	Vertikales Navigationsmenu auf der linken Seite, welches in Abhängigkeit von Nutzerrechten verschiedene Funktionsgruppen anzeigt, eigentliche Arbeitsfläche anschließend rechts daneben. Zusätzlich existieren ein javabasierter Workfloweditor (ProEd) in einem externen Fenster sowie ein XForms-Editor für die Workflowausgabe
Layout	Strikte Trennung von Funktionsauswahl und Modulinhalt, Symbolisierung der Funktionen durch kleine Icons in Baumstruktur
Intuitive Bedienbarkeit	Ja, nach kurzer Einarbeitungszeit und Verständnis der Baumstruktur
Graphische Modellierwerkzeuge	Ja, javabasierter Workfloeditor (ProEd)
Modulübergreifende Datenauswertung	Ja, Workflowbeschreibung im Mittelpunkt, aber auch alle anderen Nutzeraktivitäten werden registriert und sind über den Operator-Funktionsblock ansehbar und manipulierbar
Nutzerverwaltung	Ja, Anmeldedialog für Workflow Process Console, Auflösung der Workflow participants entsprechend Rollendefinition in Workflow entweder über Attribute oder Verzeichnisdienst (LDAP)
Aufgabenverwaltung	Innerhalb des Workflow-Editors Aufgaben frei definierbar einschließlich etwaiger Bedingungen und Übergänge
Strukturierung von Aktivitäten	Ja, Aktivitäten sind hierarchisch strukturierbar
Dokumentenverwaltung	n/a
Zeit- und Forschrittsauswertungen	Nur anhand des aktuellen Workflowstatus (ToStart, Running, ToDo, Done)
Systembenachrichtigungsfunktionen	Ja
Kommunikationsfunktionen	n/a
Weitere Groupware-Funktionen	n/a (nur Logging aller Systemaktvitäten und Nutzerinteraktionen sowie Workflowveränderungen)
Anwendungsszenarien	
Szenario 1: Spaghetti kochen	In Workflow Editor problemlos modellierbar, auch wenn die anschließende Ausführung in der Workflow-Engine nur eine Person involviert
Szenario 2: Abendessen mit Freunden	Ebenfalls als Workflow modellierbar mit Gastgeber und Gästen als entsprechenden Rollen, denen durch die Workflowengine nacheinander durchzuführende Aufgaben zugeteilt werden
Szenario 3: Diplomarbeit schreiben	Swimlane-Konzept gibt es nicht, die Einstellung der ausführenden Personen erfolgt über ein Kontextmenu, an sich kann das Anwendungsszenario aber wie gehabt durch einen Workflow im Editor abgebildet werden, auch wenn der jeweilige Rolleninhaber (z.B. Betreuer) den aktuellen Arbeitsfortschritt der Anderen (des Diplomanden) nur indirekt erkennen kann
Szenario 4: Applikationsentwurf	Auch abbildbar, aber nicht geeignet, da typische Projektmanagementfunktionen fehlen
Szenario 5: Workshop organisieren	An sich auch modellierbar, da der XPDL-Editor die nötige Untestützung bietet und ggf. aus der Arbeitsfläche herausgezoomt werden kann, die Umsetzung in der Workflow Engine ist jedoch nur bei einem fest vorgegebenen Ablauf sinnvoll
Gesamteindruck	
Hilfreiche Funktionen	Trennung verschiedener Funktionen wie Workflowdesign,

	Workflowausführung, Überwachung und Systemadministration, XForms Editor
Fehlende Funktionen	Editor sehr dialoglastig, entsprechend der XPDL-Spezifikation könnten mehr grafische Grundprimitive zur Verfügung stehen, Zwischenspeichern der XPDL-Beschreibung des Workflows und anschließendes Laden der gespeicherten Datei in Workflow-Engine könnte entfallen
Bewertung	Bonita ist ein komplettes Workflowmanagementsystem, welches von Haus aus eine Workflow-Engine und einen Workflow-Editor mitbringt, und diese nahtlos kombiniert. Die Designer- und Operatorfunktionen sind gut konzipiert und einfach zu erreichen

Produktname	Imixs IX JEE Workflow
Generelle Informationen	
Hersteller	Imixs.com
Version	5.1
Webseite	http://www.imixs.org/
Entwicklungsbeginn	12.12.2005
Letzte Aktualisierung	22.05.2007
Managementsystem-Kategorie	Workflow-Management
Plattform	Webbasiert / Editor für Eclipse
Programmiersprache	Java
Datenmodell	Proprietär (IXM-Datei, enthält XML-Daten), kann aber als XPDL exportiert werden
Addons / Plugins vorhanden	Nein
Sprachen	Englisch
Lizenz	LGPL
System	
Funktionsgruppen	In Beispielimplementierung nur die Möglichkeit, einen neuen Beispielprozess zu erstellen und zu editieren
Navigationsstruktur / Bedienkonzept	Nutzerbasierte Worklist mit aktiven Prozessen, nach Auswahl eines Prozesses rechts daneben ein Editierfeld zum Modifizieren des aktuellen Prozessstatus
Layout	Einfacher Webclient mit Listenansicht
Intuitive Bedienbarkeit	AJAX-Funktionalitäten vorhanden, Usability in Beispielanwendung aber bestmöglich unterstützt, so sind zum Beispiel die Schaltflächen zum Starten eines neuen Prozesses oder zum Aktualisieren der Prozessliste nur nach Scrollen am unteren Tabellenrand zu finden
Graphische Modellierwerkzeuge	Über Eclipse-Plugin möglich, webbasiert keine bereitgestellt
Modulübergreifende Datenauswertung	n/a
Nutzerverwaltung	In Beispielanwendung nur serverseitig realisiert, wobei Authentifizierungsdaten anschließend zur Zuordnung aktiver Workflows benutzt werden
Aufgabenverwaltung	Über Workflow-Editor können neue Aktivitäten wie gehabt erstellt werden, Grundprimitive sind Prozess, Aktivität und Übergang
Strukturierung von Aktivitäten	Nein, nur durch Prozessfluss
Dokumentenverwaltung	n/a
Zeit- und Forschrittsauswertungen	Nur anhand des aktuellen Workflowstatus
Systembenachrichtigungsfunktionen	n/a
Kommunikationsfunktionen	n/a
Weitere Groupware-Funktionen	n/a
Anwendungsszenarien	
Szenario 1: Spaghetti kochen	Als einfacher Workflow im Editor modellier- und anschließend ausführbar
Szenario 2: Abendessen mit Freunden	Ebenfalls in Workflow abbildbar
Szenario 3: Diplomarbeit schreiben	Könnte durch verschiedene Prozesse mit mehreren Aktivitäten abgebildet werden
Szenario 4: Applikationsentwurf	Zu realisierende Aufgaben könnten innerhalb verschiedener Prozesse realisiert werden (bspw. Modul fertig implementiert ja/nein), eine Einordnung in den Gesamtkontext des Projektes ist aber nicht möglich
Szenario 5: Workshop organisieren	Die Erstellung und Delegation von Aufgaben in Form von Prozessen ist generell möglich und den jeweiligen Bearbeitern zuordbar, für eine sinnvolle Kollaboration zur Umsetzung des Projektes fehlen aber weitere Funktionen
Gesamteindruck	
Hilfreiche Funktionen	AJAX-Support mit direkter Systeminteraktion, Webservice-API zur Anbindung an BPEL-Prozesse
Fehlende Funktionen	Getesteter Webclient stellte nur eine Testumgebung mit wenigen Funktionen dar, weswegen der genaue Funktionsumfang nicht erfasst werden kann. In jedem Fall ist die Trennung des Workflow-Editors von der Webapplikation nachteilig als auch das propietäre Speicherformat
Bewertung	Im BPM-Bereich mag die Workflow-Lösung von Imixs sinnvoll einsetzbar sein, wenn mehrere BPM-Applikationen über Web Services miteinander interagieren, für einfache Workflows ist das System nicht geeignet

Produktname	OSWorkflow
Generelle Informationen	
Hersteller	OpenSymphony Development Team
Version	2.8.0
Webseite	http://www.opensymphony.com/osworkflow/
Entwicklungsbeginn	2000
Letzte Aktualisierung	08.01.2006
Managementsystem-Kategorie	Workflow-Management
Plattform	Workflow-Editor webbasiert(Swing)
Programmiersprache	Java
Datenmodell	Proprietär
Addons / Plugins vorhanden	Nein
Sprachen	Englisch
Lizenz	Apache Software License
System	
Funktionsgruppen	Nur ein Workflow-Editor mit Workflow-Ansicht und Detaileingabemaske
Navigationsstruktur / Bedienkonzept	Workflows sollen sofern möglich von Hand als XML-Dokument geschrieben, über die API geladen und über den Tomcat oder einen vergleichbaren Server zur Ausführung gebracht werden.
Layout	Editor ist sehr einfach aufgebaut, enthält nur drei graphische Grundprimitive zum Aufbau eines Workflows sowie ein Eingabefenster für Attribute, Bedingungen, auszuführende Funktionen und den aktuellen Prozessstatus
Intuitive Bedienbarkeit	Bietet nur sehr wenige Funktionen, die von der konkreten Implementierung abstrahieren
Graphische Modellierwerkzeuge	Ja, einfacher Workflow Designer
Modulübergreifende Datenauswertung	n/a
Nutzerverwaltung	n/a
Aufgabenverwaltung	Im Editor steht nur eine als „New Step“ bezeichnete Aktivität zur Verfügung
Strukturierung von Aktivitäten	Nein
Dokumentenverwaltung	n/a
Zeit- und Forschrittsauswertungen	n/a
Systembenachrichtigungsfunktionen	n/a
Kommunikationsfunktionen	n/a
Weitere Groupware-Funktionen	n/a
Anwendungsszenarien	
Szenario 1: Spaghetti kochen	Als einfacher Workflow gut abbildbar
Szenario 2: Abendessen mit Freunden	Bereits aufwändiger, da ad hoc keine Nutzer/Rollenverwaltung existiert und einzelne Aktionen selbst implementiert werden müssen
Szenario 3: Diplomarbeit schreiben	Ohne längerfristige Einarbeitung in das System und Kenntnisse in Java nicht modellierbar
Szenario 4: Applikationsentwurf	Ohne längerfristige Einarbeitung in das System und Kenntnisse in Java nicht modellierbar
Szenario 5: Workshop organisieren	Ohne längerfristige Einarbeitung in das System und Kenntnisse in Java nicht modellierbar
Gesamteindruck	
Hilfreiche Funktionen	Flexibler Ansatz an sich gut und ebenso die Idee, einen graphischen Editor im ersten Schritt nicht bereitzustellen, da diese meist ohnehin nur eine Untermenge der möglichen Funktionen bereitstellen können
Fehlende Funktionen	Für die meisten typischen Funktionen, die sich in anderen Workflowmanagementsystemen fehlen, existiert keine Abstraktion, womit diese von Hand implementiert werden müssen
Bewertung	OSWorkflow ist ein Ansatz zur Bereitstellung eines Workflowmanagementsystems, der sich in erster Linie zum programmatischen Einsatz in einer Java-Entwicklungsumgebung eignet, da fortgeschrittene Programmierkenntnisse vorausgesetzt werden

Produktname	RunaWFE
Generelle Informationen	
Hersteller	Runa Consulting Group
Version	1.0
Webseite	http://wfdemo.runa.ru/wfe/
Entwicklungsbeginn	2004
Letzte Aktualisierung	2005
Managementsystem-Kategorie	Workflow-Management (Engine)
Plattform	Webbasiert
Programmiersprache	Java
Datenmodell	jPdl
Addons / Plugins vorhanden	Nein
Sprachen	Englisch, Deutsch, Französisch, Russisch
Lizenz	GPL
System	
Funktionsgruppen	Aufgaben, Prozessdefinitionen, Prozessfälle, Benutzer, System
Navigationsstruktur / Bedienkonzept	Einfaches vertikales Navigationsmenu am linken Bildschirmrand, Prozessauflisten im größeren rechten Teil der Seite
Layout	Funktional, alle wesentlichen Funktionen sind sofort erreichbar
Intuitive Bedienbarkeit	Ja
Graphische Modellierwerkzeuge	Nein (jBPM-Editor-Plugin für Eclipse verfügbar)
Modulübergreifende Datenauswertung	n/a
Nutzerverwaltung	Ja
Aufgabenverwaltung	Definition von Aktivitäten in Workflowdefinition, Start neuer Prozesse über Webapplikation
Strukturierung von Aktivitäten	Nein
Dokumentenverwaltung	n/a
Zeit- und Forschrittsauswertungen	Nur über Workflowstatus / Start- und Enddatum
Systembenachrichtigungsfunktionen	n/a
Kommunikationsfunktionen	n/a
Weitere Groupware-Funktionen	n/a
Anwendungsszenarien	
Szenario 1: Spaghetti kochen	Workflow direkt modellierbar mit verschiedenen Aktivitäten
Szenario 2: Abendessen mit Freunden	Ebenso als Prozess abbildbar, der mehrere Benutzer umfasst, denen die Benutzerrolle Gastgeber und Gast zugeordnet ist
Szenario 3: Diplomarbeit schreiben	Wie in vorausgehenden Workflowmanagementsystemen prinzipiell als Workflow mit mehreren Beteiligten modellierbar und in Workflow-Engine gut ausführbar, ein paar Funktionen aus dem Projekt- und Dokumentenmanagement sind darüber hinaus aber möglich
Szenario 4: Applikationsentwurf	Projektkontext fehlt, Abbildung einfacher Aufgaben in Workflowengine möglich, Modellierung des gesamten Anwendungsszenarios aufwändig
Szenario 5: Workshop organisieren	Für im vorausgeplante und festgelegte Organisationspläne durchaus benutzbar, Modellierung eines komplexen Workflows mit einer Vielzahl an Beteiligten sonst ad hoc schwierig
Gesamteindruck	
Hilfreiche Funktionen	Versionsmanagement für Prozesse, automatische Sprachunterstützung für mehrere Sprachen
Fehlende Funktionen	Webbasiertes Werkzeug zur Erstellung und Modifikation von Workflows
Bewertung	RunaWFE ist ein kleines und im praktischem Umfeld weniger bekanntes System, welches auf die jBoss jBpm BPM Engine setzt und eine intuitve Ausführung vorab definierter Workflows ermöglicht.

Produktname	Oryx
Generelle Informationen	
Hersteller	Martin Czuchra, Nicolas Peters, Daniel Polak, Willi Tscheschner (Hasso Plattner Institut, Universität Potsdam)
Version	0.1
Webseite	http://www.oryx-editor.org/
Entwicklungsbeginn	2006
Letzte Aktualisierung	30.07.2007
Managementsystem-Kategorie	Workflow-Management (Editor)
Plattform	Web-basiert
Programmiersprache	Javascript mit Prototype und YUI, eRDF
Datenmodell	BPMN
Addons / Plugins vorhanden	Ja
Sprachen	Englisch
Lizenz	MIT
System	
Funktionsgruppen	Modellierung von BPMN-Diagrammen mithilfe verschiedene grafischer Grundprimitive (Task, Subprocess, Gateways, Pools, Swimlanes, Artifacts, Event, Connectors)
Navigationsstruktur / Bedienkonzept	Menüleiste mit Datei- und Zwischenablagenoperationen, Shape Repository am linken Bildrand, daneben Arbeitsbereich, rechts Elgenschaftenbereich
Layout	Windows-ähnlich
Intuitive Bedienbarkeit	Ja, wobei einige Abhängigkeiten noch verbessert werden könnten. So muss erst eine Swimlane erstellt werden, in welcher anschließend Aufgaben erstellt werden können
Graphische Modellierwerkzeuge	Ja
Modulübergreifende Datenauswertung	n/a
Nutzerverwaltung	n/a
Aufgabenverwaltung	Graphische Modellierung von Prozessen durch Abfolge von Aktivitäten in BPMN
Strukturierung von Aktivitäten	Ja, Unterstützung von Subprozessen
Dokumentenverwaltung	n/a
Zeit- und Forschrittsauswertungen	n/a
Systembenachrichtigungsfunktionen	n/a
Kommunikationsfunktionen	n/a
Weitere Groupware-Funktionen	n/a
Anwendungsszenarien	
Szenario 1: Spaghetti kochen	Mithilfe der BPMN-Primitive im Oryx-Editor modellierbar
Szenario 2: Abendessen mit Freunden	Mithilfe der BPMN-Primitive im Oryx-Editor modellierbar
Szenario 3: Diplomarbeit schreiben	Mithilfe der BPMN-Primitive im Oryx-Editor modellierbar
Szenario 4: Applikationsentwurf	Mithilfe der BPMN-Primitive im Oryx-Editor modellierbar
Szenario 5: Workshop organisieren	Mithilfe der BPMN-Primitive im Oryx-Editor modellierbar
Gesamteindruck	
Hilfreiche Funktionen	Bereitstellung der BPMN-Beschreibung als eRDF direkt im HTML-Quellcode der Seite, welche beispielsweise in BPEL konvertiert und von einer entsprechenden Workflow Engine ausgeführt werden kann
Fehlende Funktionen	Oryx konzentriert sich momentan auf die Bereitstellung eines reinen Grafikwerkzeuges zur Modellierung von Business-Prozessen. Eine Gesamtlösung mit einer BPEL-Workflowengine wäre anstrebenswert. Von algorithmischer Seite momentan noch funktionelle Probleme im Microsoft Internet Explorer
Bewertung	Außer Konkurrenz im Vergleich zu den anderen Workflow-managementsystemen wird mit dem Oryx-Projekt momentan gezeigt, dass auch ein webbasierter Editor für die Business Project Modelling Notation (BPMN) realisierbar und leistungsfähig ist, wobei die Beschreibung

A.3. Verwendete RDF Schemata zur Abbildung der Projekt- und Workflowmanagementdomain

Project.rdfs

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns="http://vsr.informatik.tu-chemnitz.de/projects/2007/semproj/project#"
    xmlns:semproj="http://vsr.informatik.tu-chemnitz.de/projects/2007/semproj#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml:base="http://vsr.informatik.tu-chemnitz.de/projects/2007/semproj/project">
  <rdfs:Class rdf:ID="project">
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >a Project Item containing Project information</rdfs:comment>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Project</rdfs:label>
  </rdfs:Class>
  <rdfs:Class rdf:ID="projectrole">
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >a grouped subsumption of stakeholders</rdfs:comment>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >ProjectRole</rdfs:label>
  </rdfs:Class>
  <rdf:Property rdf:ID="title">
    <rdfs:domain rdf:resource="#project"/>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Title</rdfs:label>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >the title of the Project</rdfs:comment>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  </rdf:Property>
  <rdf:Property rdf:ID="plannedstartdate">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#date"/>
    <rdfs:domain rdf:resource="#project"/>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >the planned start date of the project</rdfs:comment>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >PlannedStartDate</rdfs:label>
  </rdf:Property>
  <rdf:Property rdf:ID="description">
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Description</rdfs:label>
    <rdfs:domain rdf:resource="#project"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >a Description of the Project content or action</rdfs:comment>
  </rdf:Property>
  <rdf:Property rdf:ID="createdby">
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >a reference to a person that created the Project</rdfs:comment>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
```

```
    >CreatedBy</rdfs:label>
    <rdfs:domain rdf:resource="#project"/>
    <rdfs:range                                    rdf:resource="http://vsr.informatik.tu-
chemnitz.de/projects/2007/semproj#user"/>
  </rdf:Property>
  <rdf:Property rdf:ID="lastmodifiedby">
    <rdfs:domain rdf:resource="#project"/>
    <rdfs:range                                    rdf:resource="http://vsr.informatik.tu-
chemnitz.de/projects/2007/semproj#user"/>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Link to the user who modified the Project last</rdfs:comment>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >LastModifiedBy</rdfs:label>
  </rdf:Property>
  <rdf:Property rdf:ID="hasmember">
    <rdfs:domain rdf:resource="#project"/>
    <rdfs:range                                    rdf:resource="http://vsr.informatik.tu-
chemnitz.de/projects/2007/semproj#user"/>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Specifies the stakeholders (members) participating on the project</rdfs:comment>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >hasMember</rdfs:label>
  </rdf:Property>
  <rdf:Property rdf:ID="hasrole">
    <rdfs:range rdf:resource="#projectrole"/>
    <rdfs:domain                                   rdf:resource="http://vsr.informatik.tu-
chemnitz.de/projects/2007/semproj#user"/>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >the reference to a group role the user belongs to</rdfs:comment>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >hasRole</rdfs:label>
  </rdf:Property>
  <rdf:Property rdf:ID="hasname">
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >hasName</rdfs:label>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#projectrole"/>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >the name of the project role group</rdfs:comment>
  </rdf:Property>
  <rdf:Property rdf:ID="actualstartdate">
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >the actual start date of the project</rdfs:comment>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >ActualStartDate</rdfs:label>
    <rdfs:domain rdf:resource="#project"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#date"/>
  </rdf:Property>
  <rdf:Property rdf:ID="createdon">
    <rdfs:domain rdf:resource="#project"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#date"/>
```

```
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >CreatedOn</rdfs:label>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >the creation time of the Project</rdfs:comment>
  </rdf:Property>
  <rdf:Property rdf:ID="status">
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Status</rdfs:label>
    <rdfs:domain rdf:resource="#project"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >the status of Project</rdfs:comment>
  </rdf:Property>
  <rdf:Property rdf:ID="ispublic">
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >isPublic</rdfs:label>
    <rdfs:domain rdf:resource="#project"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >specifies if the project information is available for public access</rdfs:comment>
  </rdf:Property>
  <rdf:Property rdf:ID="actualenddate">
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >the actual end date of the project</rdfs:comment>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >ActualEndDate</rdfs:label>
    <rdfs:domain rdf:resource="#project"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#date"/>
  </rdf:Property>
  <rdf:Property rdf:ID="plannedenddate">
    <rdfs:domain rdf:resource="#project"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#date"/>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >PlannedEndDate</rdfs:label>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >the planned end date of the project</rdfs:comment>
  </rdf:Property>
  <rdf:Property rdf:ID="lastmodifiedon">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#date"/>
    <rdfs:domain rdf:resource="#project"/>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >LastModifiedOn</rdfs:label>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Date when the related Project was modified for the last time</rdfs:comment>
  </rdf:Property>
</rdf:RDF>

<!-- Created with Protege (with OWL Plugin 3.3, Build 414)  http://protege.stanford.edu -->
```

Process.rdfs

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns="http://vsr.informatik.tu-chemnitz.de/projects/2007/semproj/process#"
    xmlns:semproj="http://vsr.informatik.tu-chemnitz.de/projects/2007/semproj#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml:base="http://vsr.informatik.tu-chemnitz.de/projects/2007/semproj/process">
  <rdfs:Class rdf:ID="process">
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >a process Item containing process information</rdfs:comment>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Process</rdfs:label>
  </rdfs:Class>
  <rdf:Property rdf:ID="title">
    <rdfs:domain rdf:resource="#process"/>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Title</rdfs:label>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >the title of the process</rdfs:comment>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  </rdf:Property>
  <rdf:Property rdf:ID="startdate">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#date"/>
    <rdfs:domain rdf:resource="#process"/>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >the start date of the process</rdfs:comment>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >StartDate</rdfs:label>
  </rdf:Property>
  <rdf:Property rdf:ID="description">
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Description</rdfs:label>
    <rdfs:domain rdf:resource="#process"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >a Description of the process content or action</rdfs:comment>
  </rdf:Property>
  <rdf:Property rdf:ID="createdby">
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >a reference to a person that created the process</rdfs:comment>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >CreatedBy</rdfs:label>
    <rdfs:domain rdf:resource="#process"/>
    <rdfs:range                                    rdf:resource="http://vsr.informatik.tu-
chemnitz.de/projects/2007/semproj#user"/>
  </rdf:Property>
  <rdf:Property rdf:ID="lastmodifiedby">
    <rdfs:domain rdf:resource="#process"/>
```

```
    <rdfs:range                                    rdf:resource="http://vsr.informatik.tu-
chemnitz.de/projects/2007/semproj#user"/>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Link to the user who modified the process last</rdfs:comment>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >LastModifiedBy</rdfs:label>
  </rdf:Property>
  <rdf:Property rdf:ID="createdon">
    <rdfs:domain rdf:resource="#process"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#date"/>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >CreatedOn</rdfs:label>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >the creation time of the process</rdfs:comment>
  </rdf:Property>
  <rdf:Property rdf:ID="status">
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Status</rdfs:label>
    <rdfs:domain rdf:resource="#process"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >the status of process</rdfs:comment>
  </rdf:Property>
  <rdf:Property rdf:ID="enddate">
    <rdfs:domain rdf:resource="#process"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#date"/>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >EndDate</rdfs:label>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >the end date of the process</rdfs:comment>
  </rdf:Property>
  <rdf:Property rdf:ID="lastmodifiedon">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#date"/>
    <rdfs:domain rdf:resource="#process"/>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >LastModifiedOn</rdfs:label>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Date when the related process was modified for the last time</rdfs:comment>
  </rdf:Property>
</rdf:RDF>

<!-- Created with Protege (with OWL Plugin 3.3, Build 414)  http://protege.stanford.edu -->
```

Activity.rdfs

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns="http://vsr.informatik.tu-chemnitz.de/projects/2007/semproj/activity#"
    xmlns:semproj="http://vsr.informatik.tu-chemnitz.de/projects/2007/semproj#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml:base="http://vsr.informatik.tu-chemnitz.de/projects/2007/semproj/activity">
  <rdfs:Class rdf:ID="activity">
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >a activity Item containing activity information</rdfs:comment>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Activity</rdfs:label>
  </rdfs:Class>
  <rdf:Property rdf:ID="title">
    <rdfs:domain rdf:resource="#activity"/>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Title</rdfs:label>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >the title of the activity</rdfs:comment>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  </rdf:Property>
  <rdf:Property rdf:ID="planneddate">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#date"/>
    <rdfs:domain rdf:resource="#activity"/>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >the planned date to execute the activity</rdfs:comment>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >PlannedDate</rdfs:label>
  </rdf:Property>
  <rdf:Property rdf:ID="description">
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Description</rdfs:label>
    <rdfs:domain rdf:resource="#activity"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >a Description of the activity content or action</rdfs:comment>
  </rdf:Property>
  <rdf:Property rdf:ID="createdby">
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >a reference to a person that created the activity</rdfs:comment>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >CreatedBy</rdfs:label>
    <rdfs:domain rdf:resource="#activity"/>
    <rdfs:range                                    rdf:resource="http://vsr.informatik.tu-
chemnitz.de/projects/2007/semproj#user"/>
  </rdf:Property>
  <rdf:Property rdf:ID="lastmodifiedby">
    <rdfs:domain rdf:resource="#activity"/>
```

```
    <rdfs:range                                    rdf:resource="http://vsr.informatik.tu-
chemnitz.de/projects/2007/semproj#user"/>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Link to the user who modified the activity last</rdfs:comment>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >LastModifiedBy</rdfs:label>
  </rdf:Property>
  <rdf:Property rdf:ID="createdon">
    <rdfs:domain rdf:resource="#activity"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#date"/>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >CreatedOn</rdfs:label>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >the creation time of the activity</rdfs:comment>
  </rdf:Property>
  <rdf:Property rdf:ID="status">
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Status</rdfs:label>
    <rdfs:domain rdf:resource="#activity"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >the status of activity</rdfs:comment>
  </rdf:Property>
  <rdf:Property rdf:ID="actualdate">
    <rdfs:domain rdf:resource="#activity"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#date"/>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >ActualDate</rdfs:label>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >the actual date when the activity was carried out</rdfs:comment>
  </rdf:Property>
  <rdf:Property rdf:ID="lastmodifiedon">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#date"/>
    <rdfs:domain rdf:resource="#activity"/>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >LastModifiedOn</rdfs:label>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Date when the related activity was modified for the last time</rdfs:comment>
  </rdf:Property>
  <rdf:Property rdf:ID="hasdocument">
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >hasDocument</rdfs:label>
    <rdfs:domain rdf:resource="#activity"/>
    <rdfs:range                                    rdf:resource="http://vsr.informatik.tu-
chemnitz.de/projects/2007/semproj#document"/>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >refers to a document that is related to the current activity</rdfs:comment>
  </rdf:Property>
</rdf:RDF>
<!-- Created with Protege (with OWL Plugin 3.3, Build 414)  http://protege.stanford.edu -->
```

Semproj.rdfs

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns="http://vsr.informatik.tu-chemnitz.de/projects/2007/semproj#"
    xmlns:project="http://vsr.informatik.tu-chemnitz.de/projects/2007/semproj/project#"
    xmlns:process="http://vsr.informatik.tu-chemnitz.de/projects/2007/semproj/process#"
    xmlns:activity="http://vsr.informatik.tu-chemnitz.de/projects/2007/semproj/activity#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml:base="http://vsr.informatik.tu-chemnitz.de/projects/2007/semproj">
  <rdfs:Class                                          rdf:about="http://vsr.informatik.tu-
chemnitz.de/projects/2007/semproj/project#project">
    <rdfs:subClassOf>
      <rdfs:Class rdf:ID="semprojobject"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="#semprojobject">
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >SemProjObject</rdfs:label>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >a SemProjObject is an object that can be part of a project or workflow description
within SemProj</rdfs:comment>
  </rdfs:Class>
 <rdfs:Class                                           rdf:about="http://vsr.informatik.tu-
chemnitz.de/projects/2007/semproj#transition">
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >the representation of a transition between two SemProjObjects</rdfs:comment>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Transition</rdfs:label>
  </rdfs:Class>
  <rdfs:Class                                          rdf:about="http://vsr.informatik.tu-
chemnitz.de/projects/2007/semproj/process#process">
    <rdfs:subClassOf rdf:resource="#semprojobject"/>
  </rdfs:Class>
  <rdfs:Class                                          rdf:about="http://vsr.informatik.tu-
chemnitz.de/projects/2007/semproj/activity#activity">
    <rdfs:subClassOf rdf:resource="#semprojobject"/>
  </rdfs:Class>
  <rdf:Property rdf:ID="positionleft">
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >PositionLeft</rdfs:label>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >position information for an item relative to the left of the screen</rdfs:comment>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#semprojobject"/>
  </rdf:Property>
  <rdf:Property rdf:ID="transitionfrom">
    <rdfs:range rdf:resource="#semprojobject"/>
    <rdfs:domain                                    rdf:resource="http://vsr.informatik.tu-
chemnitz.de/projects/2007/semproj#transition"/>
```

```
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >TransitionFromEntity</rdfs:label>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >SemProjObject where the transition starts</rdfs:comment>
  </rdf:Property>
  <rdf:Property rdf:ID="positiontop">
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >PositionTop</rdfs:label>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >position information for an item relative to the top of the screen</rdfs:comment>
    <rdfs:domain rdf:resource="#semprojobject"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  </rdf:Property>
  <rdf:Property rdf:ID="parententity">
    <rdfs:range rdf:resource="#semprojobject"/>
    <rdfs:domain rdf:resource="#semprojobject"/>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >the super SemProjObject containing the current SemProjObject, if any</rdfs:comment>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >ParentEntity</rdfs:label>
  </rdf:Property>
  <rdf:Property rdf:ID="subentity">
    <rdfs:range rdf:resource="#semprojobject"/>
    <rdfs:domain rdf:resource="#semprojobject"/>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >a subordinated SemProjObject within the current SemProjObject, if any</rdfs:comment>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >ContainsEntity</rdfs:label>
  </rdf:Property>
  <rdf:Property rdf:ID="join">
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >join operation before the item</rdfs:comment>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Join</rdfs:label>
    <rdfs:domain rdf:resource="#semprojobject"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  </rdf:Property>
  <rdf:Property rdf:ID="transitionto">
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >TransitionToEntity</rdfs:label>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >SemProjObject where the transition ends</rdfs:comment>
    <rdfs:domain                                    rdf:resource="http://vsr.informatik.tu-
chemnitz.de/projects/2007/semproj#transition"/>
    <rdfs:range rdf:resource="#semprojobject"/>
  </rdf:Property>
  <rdf:Property rdf:ID="split">
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >split operation after the item</rdfs:comment>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#semprojobject"/>
```

```
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Split</rdfs:label>
  </rdf:Property>
  <rdf:Property rdf:ID="hasright">
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >the right owned by the group role</rdfs:comment>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >hasRight</rdfs:label>
    <rdfs:domain                                    rdf:resource="http://vsr.informatik.tu-
chemnitz.de/projects/2007/semproj/project#projectrole"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  </rdf:Property>
  <rdf:Property rdf:ID="hasindividualright">
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >a right specific to a single user</rdfs:comment>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >hasSpecificRight</rdfs:label>
    <rdfs:domain                                    rdf:resource="http://vsr.informatik.tu-
chemnitz.de/projects/2007/semproj#user"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  </rdf:Property>
  <rdf:Property rdf:ID="transition">
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >specifies a transition of type semproj:transition between two subentities within the
current SemProjObject</rdfs:comment>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >hasTransition</rdfs:label>
    <rdfs:domain rdf:resource="#semprojobject"/>
    <rdfs:range                                     rdf:resource="http://vsr.informatik.tu-
chemnitz.de/projects/2007/semproj#transition"/>
  </rdf:Property>
  <rdfs:Class rdf:ID="user">
  </rdfs:Class>
  <rdfs:Class rdf:ID="document">
  </rdfs:Class>
</rdf:RDF>

<!-- Created with Protege (with OWL Plugin 3.3, Build 414)  http://protege.stanford.edu -->
```

A.4. Analyse und Spezifikation der zu entwickelnden Anwendung

Analyse und Spezifikation des Systems

Semantic Web – basiertes System zur Unterstützung von Workflow- und Projektmanagement

Im Rahmen einer Diplomarbeit an der
Professur für Verteilte und Selbstorganisierende Rechnersysteme
Fakultät für Informatik
Technische Universität Chemnitz

erarbeitet von: André Langer
Stand: v1.3, 30.10.2007

Inhaltsverzeichnis

1. SPEZIFIKATION DER FUNKTIONELLEN ANFORDERUNGEN 35

1.1. PRODUKTBESCHREIBUNG 35

1.2. NUTZUNGSUMGEBUNG 37

1.3. NUTZERKLASSEN 37

1.4. KRITERIENKATALOG 38

1.4.1. Pflichtkriterien 38

1.4.2. Kannkriterien 41

1.5. ANWENDUNGSFALL-MODELLIERUNG 42

1.6. DEFINITION DER NUTZERSCHNITTSTELLE 43

1.7. DATENGRUNDLAGE 50

2. SPEZIFIKATION DER OPERATIONELLEN ANFORDERUNGEN 52

2.1. OPERATIONELLE ANFORDERUNGEN AN DIE DATEN 52

2.2. OPERATIONELLE ANFORDERUNGEN AN DIE PROZESSE 54

3. SPEZIFIKATION WICHTIGER QUALITÄTSANFORDERUNGEN 56

3.1. BENUTZBARKEIT 56

3.2. ZUVERLÄSSIGKEIT 56

3.3. INTEGRITÄT 56

3.4. FLEXIBILITÄT 57

3.5. PORTABILITÄT 57

4. BASISMASCHINE UND ENTWICKLUNGSUMGEBUNG 58

4.1. HARDWARE- UND SOFTWAREANFORDERUNGEN 58

4.2. ENTWICKLUNGSUMGEBUNG 58

5. SYSTEMARCHITEKTUR 59

6. KLASSENSPEZIFIKATION 60

0. Änderungskontrolle

Version	Datum	Autor	Qualitätssicherung	Status
v1.0	16.09.2007	Langer	Grundgerüst Spezifikation	Out-of-date
v1.1	20.09.2007	Langer	Erster Draft nach Treffen vom 19.09.07	Out-of-date
v1.2	16.10.2007	Langer	Zweiter Draft	Out-of-date
v1.3	30.10.2007	Langer	Fertigstellung Spezifikation	

1. Spezifikation der funktionellen Anforderungen

1.1. Produktbeschreibung

Zu entwickeln ist eine Webapplikation, welche die kollaborative Arbeit an unterschiedlichen Projekten ermöglicht. Die Anwendungsdomain der Projekte ist dabei nicht eingeschränkt – das System soll für verschiedenste Anwendungsbereiche benutzbar sein. Dennoch ist in dem zu entwickelnden System ein wissenschaftlicher Prototyp zu sehen, der sich an den Anforderungen in Projekt – und Forschungsgruppen orientiert. Der unternehmerische Aspekt des Projektmanagements, wo speziell die Verwaltung verschiedener anderer Ressourcen wie Arbeitszeit oder Kosten interessiert ist, trete unter dieser Zielsetzung in den Hintergrund. Das Gesamtsystem zerfalle in mehrere Teilsysteme, welche nachfolgend beschrieben sind:

- Anmeldedialog

Zur Interaktion mit dem System müssen sich Benutzer über einen Anmeldedialog authentifizieren. Bei Eingabe einer gültigen Benutzername/Passwort-Kombination wird der Nutzer auf eine Übersichtsseite weitergeleitet.

- Übersichtsseite im Backend

Auf der Übersichtsseite werden in einer Liste alle laufenden Projekte dargestellt, an denen der angemeldete Nutzer momentan beteiligt ist. Neben dem Projekttitel sowie Start- und Enddatum des Projektes wird zusätzlich der aktuelle Projektstatus sowie die Rolle des Benutzers in diesem Projekt (siehe Projekteditor) angezeigt. Darunter befindet sich eine weitere Übersicht über die von dem Nutzer als nächstes zu erledigenden Aufgaben in den jeweiligen Projekten. Neben der Aufgabenbezeichnung wird diese zunächst in den zugehörigen Prozess oder Teilprojekt eingeordnet, in welchem diese enthalten ist und anschließend in das Gesamtprojekt, welchem diese entstammt. Daneben findet der Nutzer eine Möglichkeit, die Aufgabe als erledigt zu markieren.

- Projekteditor

Sowohl die Modellierung neuer Projekte, als auch die Durchführung und Überwachung existierender Projekte soll mit dem zu entwickelnden System ermöglicht werden. Die Erstellung eines Projektes geschieht dabei in einem Projekteditor mit grafischer Benutzeroberfläche, bei dem besonderer Fokus ist auf eine intuitive Bedienbarkeit zu legen ist. Projekte lassen sich darin hierarchisch untergliedern in einzelne Unterprojekte („Projektphasen“), welche wiederum Projekte sein können und aus einzelnen Tätigkeitsanweisungen bestehen.

Diese Tätigkeitsanweisungen („Workflows“) können strukturiert (in Form von Prozessen) oder elementar (in Form einzelner Aktivitäten) sein. Prozesse können dabei wiederum aus anderen Prozessen oder einzelnen Aktivitäten bestehen. Zwischen Projekten und Prozessen kann eine Ordnungsrelation hergestellt werden in der Form, dass ein bestimmtes Teilprojekt erst mit Fertigstellung eines vorhergehenden Teilprojektes begonnen werden kann oder eine Tätigkeit erst ausgeführt werden kann, wenn eine vorhergehende Tätigkeit erfolgreich durchgeführt wurde. Diese Abhängigkeit muss nicht linear sein, sondern kann sich auf mehrere Instanzen beziehen, die vorausgehen (Join) oder nachfolgen (Split), vgl. dazu Workflow Patterns nach Prof. Wil van der Aalst, Universität Eindhoven, www.workflowpatterns.com. Existieren keine Zustandsübergänge zwischen zwei Objekten in dem Projektgraph (mehrere disjunkte Graphen auf einer Darstellungsebene), so sind diese unabhängig voneinander durchzuführen, um das übergeordnete Projekt erfolgreich abzuschließen. Eine zusammenhängende Folge aus Projekten, Prozessen oder Aktivitäten beginnt bei demjenigen Objekt, welches keinen Vorgänger besitzt und ist erfolgreich abgeschlossen, wenn das Objekt abgeschlossen wurde, welches keine ausgehende Kante zu einem nachfolgenden Teilprojekt/prozess/aktivität besitzt. Um ein Projekt, Prozess oder eine Aktivität auszuführen, kann der Nutzer den Status des betreffenden Objektes ändern, sofern er an der Durchführung von dessen beteiligt ist und die bezeichnete Tätigkeit zur Ausführung steht. Als Status ist bei Projekten und Prozessen eine der Eigenschaften „nicht begonnen“, „laufend“, „abgeschlossen“ bzw. „abgebrochen“ anzugeben, bei Aktivitäten „offen“ oder „durchgeführt“.

Jedem Objekt in dem Projekteditor können bestimmte Eigenschaftswerte zugeordnet werden. Neben Datumsangaben zur Durchführung der jeweiligen Tätigkeit sind dies ein Titel der Tätigkeit, eine Kurzbeschreibung, sowie die Zugriffsberechtigungen für daran beteiligte Nutzer. Jedem Projekt kann dabei eine beliebige Anzahl an Mitarbeitern zugeordnet werden, welche gewisse Rollen inne haben. Die Bezeichnung dieser Rollen kann frei gewählt werden (z.B. Projektleiter, Programmierer, Layouter, Betreuer) und dient der Gruppierung mehrerer Nutzer, um darüber gemeinsame Zuständigkeiten festlegen zu können. Die Zugriffsberechtigungen legen fest, wer die Details der nächsten Hierarchieebene lesen darf, aktuelle Projekteigenschaften verändern und wer die jeweilige Tätigkeit auszuführen hat und damit den Status der Erledigung verändern darf. Daneben können ebenso feingranular Systemmitteilungen zugeordnet werden, wer bei einer bestimmten Aktion benachrichtigt wird. Aktivitäten können darüber hinaus eine beliebige Anzahl an Dokumenten enthalten, welche entweder physische Dateien darstellen, auf andere Ressourcen verweisen („Linkliste“) oder direkt als Fließtext eingegeben werden können („Wiki“-Konzept).

- Nutzerverwaltung

Damit sich ein Nutzer an dem System anmelden kann, müssen zunächst nur ein systemweit eindeutiger Benutzername sowie ein Identitätsprovider angegeben werden, über welchen die eingegebenen Benutzeranmeldedaten überprüft werden können. Nach erfolgter Anmeldung erhält der Benutzer eine eindeutige ID zugeordnet sowie gewisse Rechte.

Standardbenutzer dürfen an bestehenden Projekten teilnehmen, Projektleiter dürfen zusätzlich neue Projekte anlegen als auch neue Nutzer dem System hinzufügen und Administratoren dürfen darüber hinaus alle dem System bekannten Benutzer und Projekte verwalten (auch die, an denen sie nicht beteiligt sind).

Jeder Nutzer erhält unter der vergebenen Benutzer-ID eine eigene Benutzerseite zugeordnet, über welche er weitere Kontakdaten wie seinen vollständigen Namen, Emailadresse, Anschrift und ein Bild hinterlegen kann. Das Projektmanagementsystem kann diese Daten nutzen, um beispielsweise bei Änderungen in den Projekten Systemmitteilungen an die angegebene Emailadresse zu versenden.

- Versionierung

Eine Anforderung besteht darin, dass alle eingegebenen Daten in einen vorherigen Zustand zurückversetzt werden können oder sich dieser ansehen lässt. Dies umfasst sowohl die einzelnen Nutzerseiten, als auch Projekt- und Prozessbeschreibungen als auch einzelne zugeordnete Dokumente.

- Import

Weiterhin muss die Möglichkeit bestehen, bereits existierende Projekt- oder Prozessbeschreibungen in neu anzulegende Projekte und Prozesse zu übernehmen. Bei Beschreibungen, die mithilfe des systemintegrierten Projekteditors erstellt worden sind, kann dies beispielsweise durch Angabe des Titels der zu importierenden Instanz geschehen, woraufhin alle untergeordneten Projekte, Prozesse, Aktivitäten und Dokumente kopiert und dem neuen Projekt zugeordnet werden. Etwaige Statusinformationen aus den früher durchgeführten Projektteilen werden dabei zurückgesetzt. Nutzerrollen und Nutzerrechte werden gelöscht und müssen durch Angabe der Mitarbeiter an dem neu erstellten Projekt erneut eingetragen werden.

1.2. Nutzungsumgebung

Die zu entwickelnde Webapplikation stelle keine besonderen Anforderungen an die Systemplattform des Clients und soll auf einem handelsüblichen Rechner unabhängig vom verwendeten Betriebssystem in einem Browser neuerer Generation lauffähig sein (Internet Explorer 6.0 oder höher, Mozilla und alle anderen auf der Gecko-Engine basierenden Browser ab Version 1.0, Opera ab Version 7.6 u.a.).

1.3. Nutzerklassen

Das System soll prinzipiell von Jedem ohne großen Zeitaufwand oder Schulung benutzbar sein, der an einem Projekt beteiligt ist oder ein eigenes Projekt durchführen möchte. Vom System werden drei Nutzerklassen entschieden, wie in 1.1. dargestellt. Normaler Benutzer, Projektleiter und Administrator, wobei Nutzerklasse 2 und 3 eine jeweils die Rechte der vorhergehenden Nutzerklasse umfassen und nur erweiterte Zugriffsmöglichkeiten bereit gestellt bekommen.

1.4. Kriterienkatalog

1.4.1. Pflichtkriterien

In Tabelle 1 sind alle Funktionen und Merkmale des Systems aufgeführt, welche erfüllt werden müssen („Pflichtkriterien“).

ID	Funktion/Merkmal
/L01/	Nur angemeldete Benutzer können das System benutzen und eine Manipulation an bestehenden Projekten ausführen
/L02/	Zur Anmeldung ist ein Benutzername und Passwort anzugeben, welches von einem Identitätsprovider geprüft wird
/L04/	Angemeldete Nutzer erhalten eine eindeutige ID, welche bei jeder Wiederanmeldung erhalten bleibt
/L05/	Angemeldeten Nutzern wird eine Systemrolle zugeordnet, wodurch der Zugriff auf bestimmte Systemfunktionen eingeschränkt wird
/N01/	Angemeldete Nutzer mit entsprechender Berechtigung können neue Nutzer anlegen, welche das System zukünftig benutzen können
/N02/	Angemeldete Nutzer mit entsprechender Berechtigung können die Zugangsbeschreibung anderer Nutzer verändern oder wieder aufheben
/N03/	Jeder Nutzer erhält unter der vergebenen NutzerID eine eigene Nutzerseite
/N04/	Auf der Nutzerseite können persönliche Informationen und Kontaktdaten eingegeben und verändert werden
/N05/	Der Nutzer kann ein Bild hochladen
/N06/	Die eingegebenen Nutzerdaten sind mit semantischen Annotationen versehen
/N07/	Die Nutzerseite ist unter einer eigenständigen URL aufrufbar
/N08/	Wird die Nutzerseite ohne Authentifizierung direkt aufgerufen, so sind die darin enthaltenen Informationen auslesbar, aber nicht veränderbar
/P01/	Es können neue Projekte angelegt werden
/P02/	Ein Projekt hat einen Projekttitel, dieser kann allerdings leer sein
/P03/	Ein Projekt muss ein Start- und Enddatum besitzen
/P04/	Einem Projekt können Mitarbeiter anhand des Benutzernamens zugeordnet werden, welche an diesem Projekt mitarbeiten
/P05/	Mitarbeiter können nach Rollen gruppiert werden
/P06/	Über den Mitarbeiternamen oder die Rollenbezeichnung können Zugriffs- und Benachrichtigungsrechte vergeben werden
/P07/	Ein Projekt kann sich in weitere Unterprojekte oder darin enthaltene Prozesse untergliedern
/P08/	Zwischen Projekten oder Projekten und Prozessen können gerichtete Übergänge definiert

	werden, welche eine Abhängigkeit darstellen
/P09/	Hat ein Projekt mehr als eine ein- oder ausgehende Kante muss eine Eigenschaft existieren, die angibt, ob beide oder mindestens eins der angegebenen Objekte ausgeführt werden muss
/P10/	Der Projektfortschritt wird farblich hervorgehoben
/P11/	Der Projektfortschritt basiert auf der Anzahl der zu erledigenden Unteraufgaben und dem zur Verfügung stehenden Zeitrahmen („Deadline“)
/P12/	Die eingegebenen Projektdaten sind mit semantischen Annotationen versehen
/P13/	Jede Projektseite ist unter einer eigenständigen URL aufrufbar
/P14/	Wird die Projektseite ohne Authentifizierung direkt aufgerufen, so sind die darin enthaltenen Informationen maschinell auslesbar, aber nicht veränderbar
/Z01/	Ein Prozess kann nur einem übergeordneten Projekt oder übergeordneten Prozess zugeordnet werden
/Z02/	Ein Prozess kann nicht oberstes Element in der Projekthierarchie sein, sondern muss zumindest ein übergeordnetes Projekt als Container besitzen
/Z03/	Ein Prozess hat einen Titel, dieser kann allerdings leer sein
/Z04/	Ein Prozess kann ein Start- und Enddatum haben
/Z05/	Sind ein Start- und Enddatum angegeben, so müssen diese innerhalb des angegebenen, übergeordneten Projektzeitraums liegen
/Z06/	Ein Prozess kann durch Projektmitarbeiter durchgeführt werden, die von dem nächstübergeordneten Projekt geerbt werden
/Z07/	Über den Mitarbeiternamen oder die Rollenbezeichnung können Zugriffs- und Benachrichtigungsrechte vergeben werden
/Z08/	Ein Prozess kann sich in weitere Subprozesse oder darin enthaltene elementare Aktivitäten untergliedern
/Z09/	Zwischen Prozessen oder Prozessen und Aktivitäten können gerichtete Übergänge definiert werden, welche eine Abhängigkeit darstellen
/Z10/	Hat ein Prozess mehr als eine ein- oder ausgehende Kante muss eine Eigenschaft existieren, die angibt, ob beide oder mindestens eins der angegebenen Objekte ausgeführt werden muss
/Z11/	Der Prozessortschritt wird farblich hervorgehoben
/Z12/	Der Prozessfortschritt basiert auf der Anzahl der zu erledigenden Unteraufgaben und dem zur Verfügung stehenden Zeitrahmen („Deadline“)
/Z13/	Die eingegebenen Prozessdaten sind mit semantischen Annotationen versehen
/Z14/	Jede Prozessseite ist unter einer eigenständigen URL aufrufbar

/Z15/	Wird die Prozessseite ohne Authentifizierung direkt aufgerufen, so sind die darin enthaltenen Informationen maschinell auslesbar, aber nicht veränderbar
/A01/	Ein Aktivität kann nur einem übergeordneten Prozess zugeordnet werden
/A02/	Eine Aktivität kann nicht oberstes Element in der Projekthierarchie sein, sondern muss sich in einem Prozess als Containerelement befinden
/A03/	Eine Aktivität hat einen Titel, dieser kann allerdings leer sein
/A04/	Eine Aktivität kann ein Durchführungsdatum besitzen
/A05/	Ist ein Durchführungsdatum angegeben, so muss dieses innerhalb des angegebenen, übergeordneten Projektzeitraums liegen
/A06/	Eine Aktivität kann durch Projektmitarbeiter durchgeführt werden, die von dem nächstübergeordneten Projekt geerbt werden
/A07/	Über den Mitarbeiternamen oder die Rollenbezeichnung können Zugriffs- und Benachrichtigungsrechte vergeben werden
/A08/	Einer Aktivität können Dokumente zugeordnet werden, entweder mit Beschreibungscharakter oder als Durchführungsresultat der Aktivität
/A09/	Zwischen Aktivitäten können gerichtete Übergänge definiert werden, welche eine Abhängigkeit darstellen
/A10/	Hat eine Aktivität mehr als eine ein- oder ausgehende Kante muss eine Eigenschaft existieren, die angibt, ob beide oder mindestens eins der angegebenen Objekte ausgeführt werden muss
/A11/	Die eingegebenen Aktivitätsdaten sind mit semantischen Annotationen versehen
/A12/	Jede Aktivitätsseite ist unter einer eigenständigen URL aufrufbar
/A13/	Wird die Aktivitätsseite ohne Authentifizierung direkt aufgerufen, so sind die darin enthaltenen Informationen maschinell auslesbar, aber nicht veränderbar
/D01/	Dokumente können nur Aktivitäten zugeordnet werden
/D02/	Dokumente können einen Dokumenttitel und einen Informationstext besitzen
/D03/	Dokumente enthalten entweder reinen Text, einen Verweis auf eine andere Webressource oder verweisen auf ein anderes Dokument
/D04/	Externe Dokumente können direkt hochgeladen und lokal referenziert werden
/D05/	Die eingegebenen Dokumentdaten sind mit semantischen Annotationen versehen
/D06/	Jede Dokumentseite ist unter einer eigenständigen URL aufrufbar
/D07/	Wird die Dokumentseite ohne Authentifizierung direkt aufgerufen, so sind die darin enthaltenen Informationen maschinell auslesbar, aber nicht veränderbar
/U01/	Es gibt eine Übersichtsseite, auf der in Abhängigkeit vom angemeldeten Nutzer dessen laufende Projekte und zu erledigenden Aufgaben dargestellt werden
/U02/	Zur Ermittlung dieser Daten existiert ein Agent, welcher die in den einzelnen Projekt-, Prozess-, Aktivitäts-, und Nutzerseiten eingebetteten Informationen auslesen und kombi-

	nieren kann
/U03/	Es werden alle Projekte auf der Übersichtsseite angezeigt, deren Status nicht abgeschlossen ist oder deren Enddatum nicht länger als eine Woche zurückliegt
/U04/	Es werden nur diejenigen zu erledigenden Aufgaben (Aktivitäten) aufgelistet, welche in den Projekten als nächstes anstehen und von dem betreffenden Nutzer auszuführen sind
/U05/	Zeiträume und Status werden in der Übersicht farblich hervorgehoben
/U06/	Bezüglich der Projekte wird neben Projekttitel, Zeitraum und Status die Rolle des Mitarbeiters in diesem Projekt aufgeführt
/U07/	Bezüglich der zu erledigenden Aktivitäten wird neben der Aktivitätsbezeichnung (Titel) der Name des übergeordneten Prozesses oder Projekts aufgeführt, als auch der Name des Gesamtprojekts
/U08/	Übergeordnete Projekte und Prozesse als auch das betrachtete Objekt sind zu der entsprechenden Seite im Projekteditor verlinkt
/U09/	Der Status von Aktivitäten kann direkt auf der Übersichtsseite geändert werden („Aufgabe erledigt“)
/U10/	Bei einer Änderung des Systemzustands (Zugriff auf Objekt, Statusänderung, Eigenschaftenänderung) existiert eine Möglichkeit, die daran beteiligten Mitarbeiter zu informieren

Tabelle 1: Pflichtkriterien

1.4.2. Kannkriterien

In Tabelle 2 sind alle Funktionen und Merkmale des Systems aufgeführt, welche bei ausreichend Zeit zusätzlich erfüllt werden können oder deren Erfüllung in Zukunft denkbar ist. („Kannkriterien“)

ID	Funktion/Merkmal
/K01/	Die Nutzerinformationen der Mitarbeiter an einem Projekt werden gesammelt auf einer Übersichtsseite mit semantischen Annotationen dargestellt und können komplett exportiert oder durch andere Anwendungen verarbeitet werden.
/K02/	Es wird eine Exportfunktion angeboten, welche die verteilten Prozessinformationen in den Projekt-, Prozess- und Aktivitätsdateien als XPDL-Beschreibung ausgibt, worin die Zuordnung verschiedener Mitarbeiter zu einzelnen Prozessen und Aktivitäten in Swimlanes transformiert wird
/K03/	Zusätzlich zu den Objekteigenschaften wird in den Objektdateien der Zugriff registriert und mit semantischen Annotationen versehen, wodurch auf der Übersichtsseite Informationen über die letzten Aktivitäten in den eigenen Projekten extrahiert werden können
/K04/	Die Liste zu erledigender Aktivitäten auf der Übersichtsseite kann als ToDo-List gespei-

	chert und unabhängig von dem Projektmanagementsystem durch einen integrierten Serviceaufruf benutzt werden
/K05/	Alle Termindaten eines Projektes werden extrahiert und in einem Kalender übersichtlich angezeigt
/K06/	Basierend auf den Projekt-, Prozess- und Aktivitätsinformationen wird die Generierung eines Gantt-Diagramms ermöglicht
/K07/	Es wird basierend auf den semantischen Informationen eine textuelle Durchführungsbeschreibung generiert, welche Informationen über den Aufbau und die Art der Durchführung eines Projektes enthält, die ausgedruckt und von einem Menschen gelesen und verstanden werden kann.

Tabelle 2: Kann-Kriterien

1.5. Anwendungsfall-Modellierung

Zur Ableitung der Use Cases soll ein fiktives, aber realitätsnahes Anwendungsszenario dienen, welches in Abbildung 1 veranschaulicht wird.

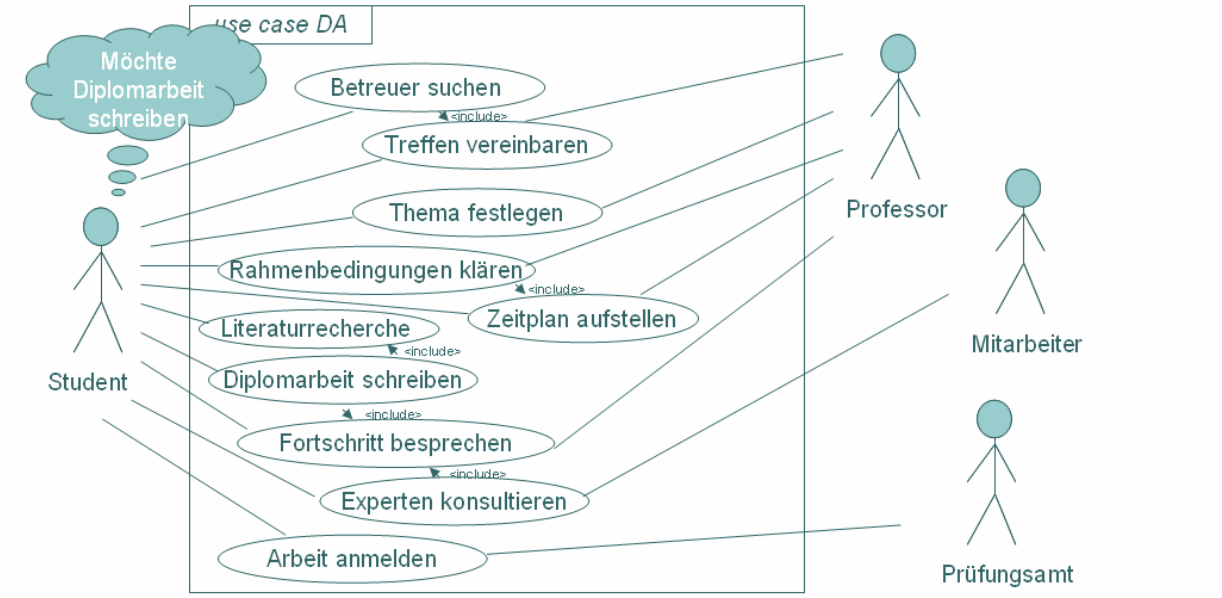


Abbildung 1: Mögliches Anwendungsszenario „Diplomarbeit schreiben“

Zu entwickeln ist ein kollaboratives System, in dem alle Benutzer von Grund auf erst einmal gleich gestellt sind und die gleichen Rechte und Möglichkeiten besitzen.

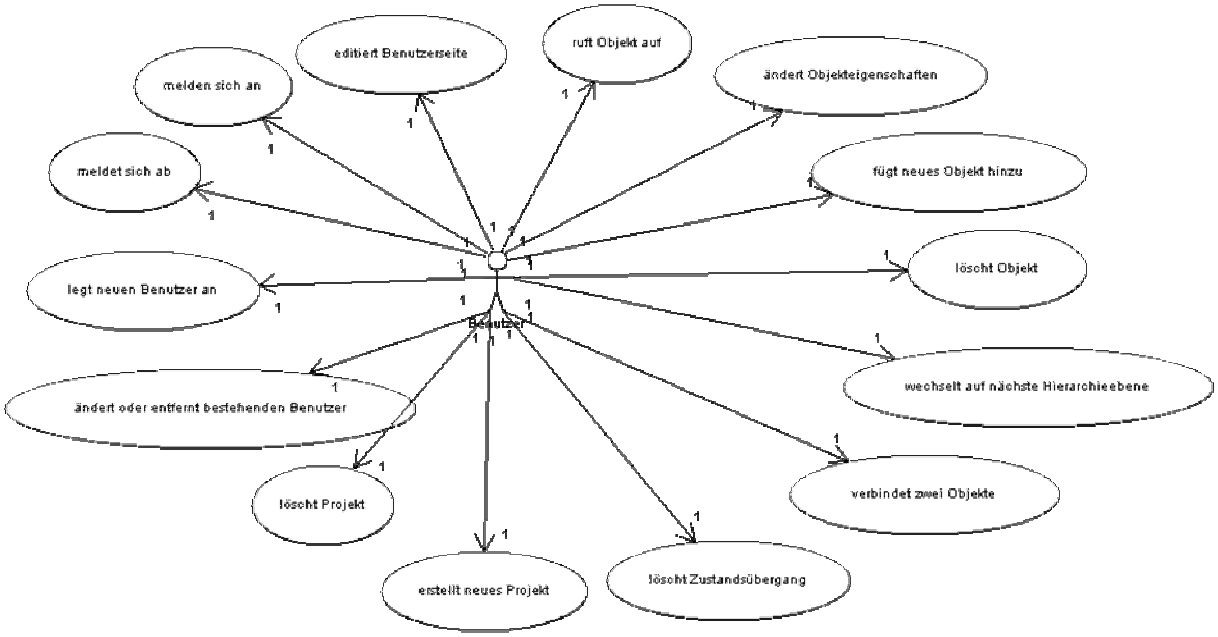


Abbildung 2: Use Case Diagramm

Im Use Case Diagramm in Abbildung zwei ist der Benutzer lediglich dahingehend zu differenzieren, das er nur Benutzer und Projekte anlegen oder löschen darf, wenn er die Rolle eines Projektmanagers oder Administrators inne hat. Die Use Case – Diagramme dieser beiden Benutzertypen sind identisch, wobei ein Administrator alle Projekte verwalten kann, ein Projektleiter nur seine eigenen. Im Use Case Diagramm sind aus Gründen der Einfachheit weiterhin unter dem Begriff „Objekt“ die Konzepte Projekt, Prozess, Aktivität und Dokument zusammengefasst.

1.6. Definition der Nutzerschnittstelle

- Fenstergröße

Die Größe der dargestellten Layoutskizzen ist nicht maßstabsähnlich, die Farbgebung entspricht nicht dem fertigen Produkt.

Die zu entwickelnde Anwendung ist als Webapplikation zu realisieren, welche in einem Browser neuerer Generation lauffähig ist und in diesen dargestellt werden kann. Unter diesen Voraussetzungen unterliegt die Anwendung allen Anforderungen und Möglichkeiten einer Webseite in einem Internetbrowser-Programm. Die Größe des Hauptfensters ist auf eine Größe von 1000x520 Pixel angepasst, was einer Monitorauflösung von 1024x768px abzüglich der Navigationsflächen des Browserfensters entspricht.

Das Hauptfenster ist in seiner Größe änderbar, wodurch die dargestellten Inhalte relativ dazu verschoben werden und weiterhin sichtbar bleiben. Alle weiteren Positionierungselemente innerhalb des Browserfensters besitzen eine feste Breite und können nicht weiter skaliert werden.

- Layout

Das Layout des Workflow- und Projektmanagementsystems ist einfach und schlicht gehalten. Im Vordergrund steht eine intuitive Bedienbarkeit, welche durch das Layout unterstützt werden soll. Hintergrundfarbe sei beispielsweise weiß, die Standardtextfarbe schwarz, wobei sich weitere Layoutelemente in unterschiedlichen Grautönen finden. Aktuelle Arbeitsbereiche und wichtige Systemhinweise werden in einer entsprechenden Signalfarbe auffallender dargestellt. Die Webseite ist dreigeteilt. Während sich am linken Bildschirmrand eine vertikale Navigationsleiste mit einem Schnellzugriff auf die wichtigsten Systemfunktionen findet, befindet sich über dem eigentlichen Inhaltsbereich eine Kopfleiste mit den wichtigsten Nutzer- und Organisationsinformationen. Über dem Inhaltsbereich findet sich weiterhin eine Kurznavigation, welche den aktuellen Navigationspfad in der Menuhierarchie darstellt. Das gesamte Layout wird als Masterpage umgesetzt, kann also relativ einfach an andere Gegebenheiten angepasst werden.

- Ein- und Ausgabegeräte

Zur Interaktion mit der Webapplikation sind eine Tastatur und Maus nötig. Die Ausgabe erfolgt grundsätzlich über den Monitor, der eine Mindestauflösung von 800x600px haben sollte.

- Layoutentwurf

Nachfolgend finden sich einzelne Screendesigns der zu entwickelnden Anwendung mit einer jeweiligen Kurzbeschreibung des dargestellten Sachverhalts

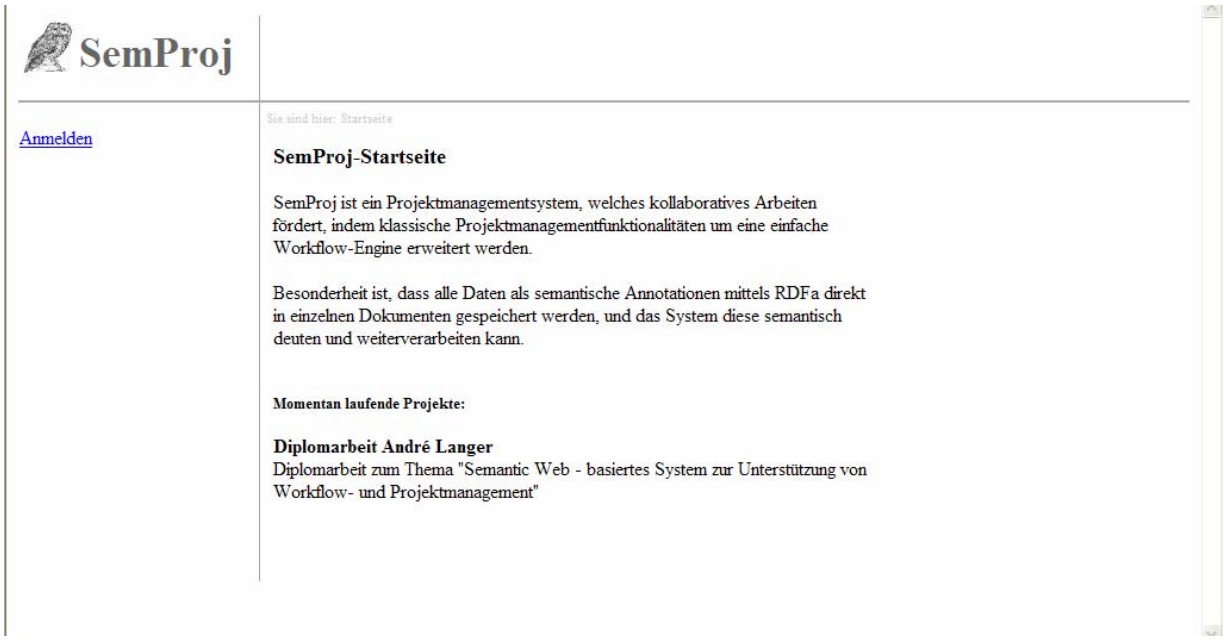

SemProj
Anmelden
Sie sind hier: Startseite
SemProj-Startseite
SemProj ist ein Projektmanagementsystem, welches kollaboratives Arbeiten fördert, indem klassische Projektmanagementfunktionalitäten um eine einfache Workflow-Engine erweitert werden.
Besonderheit ist, dass alle Daten als semantische Annotationen mittels RDFa direkt in einzelnen Dokumenten gespeichert werden, und das System diese semantisch deuten und weiterverarbeiten kann.
Momentan laufende Projekte:
Diplomarbeit André Langer
Diplomarbeit zum Thema "Semantic Web - basiertes System zur Unterstützung von Workflow- und Projektmanagement"

Abbildung 3: Startseite des Projektmanagementsystems. Neben einem Header-Bereich findet sich links klassisch das Navigationsmenu, rechts daneben der eigentliche Inhaltsbereich. Direkt darüber ist eine Kurznavigation erkennbar, welche dem Nutzer Informationen über die aktuelle Position in der Navigationshierarchie liefert

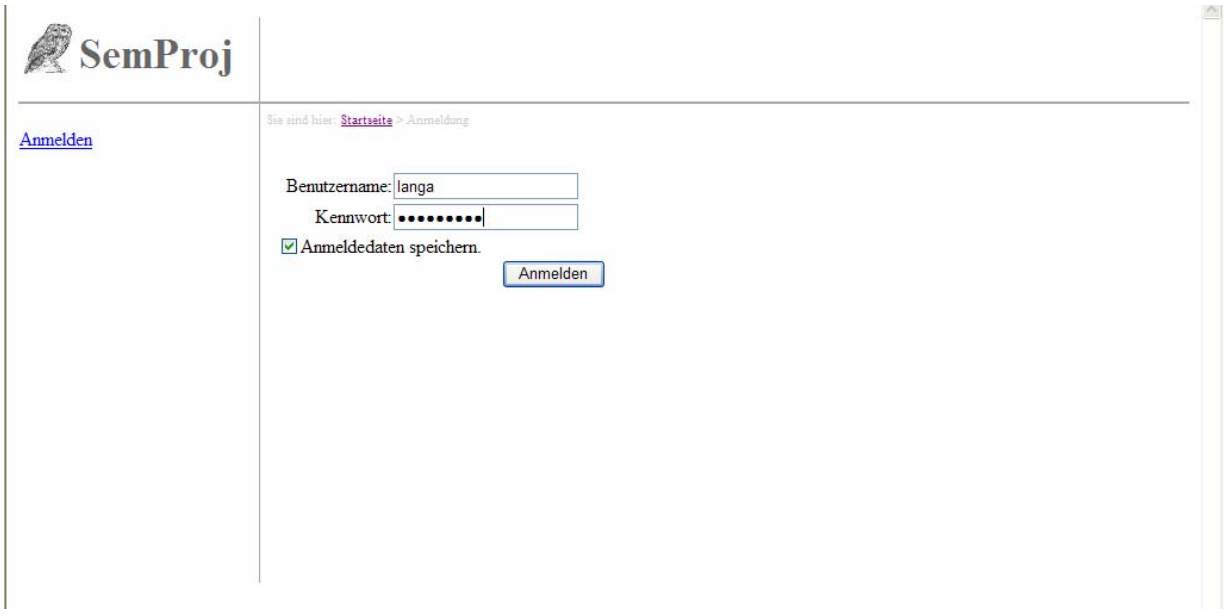


Abbildung 4: Anmeldedialog mit zwei Eingabefeldern und einer Absenden-Schaltfläche

SemProj

TECHNISCHE UNIVERSITÄT CHEMNITZ

Hallo langa, herzlich Willkommen bei SemProj

Übersichtsseite
Neues Projekt
Mein Konto
Administration

Abmelden

Sie sind hier: Startseite > Mein Konto

Profil von langa (Nutzer-ID 1)

André Langer
TU Chemnitz
Diplomand
andre.langer@informatik.tu-chemnitz.de
langer_andre@gmx.de

Dienstanschrift
Straße der Nationen 62
09111 Chemnitz Deutschland

Privatadresse
Wartburgstraße 22
09126 Chemnitz Deutschland

Tel.: 0371585115
Tel.: 01723650726

Abbildung 5: Beispielhafte Nutzerübersichtsseite mit einem hochgeladenen Bild. Alle dargestellten Inhalte sind im XHTML-Quellcode der Seite mit semantischen Annotationen versehen

SemProj

TECHNISCHE UNIVERSITÄT CHEMNITZ

Hallo langa, herzlich Willkommen bei SemProj

Übersichtsseite
Neues Projekt
Mein Konto
Administration

Abmelden

Sie sind hier: Startseite > Übersichtsseite

Meine Übersichtsseite

Laufende Projekte, an denen ich beteiligt bin

Projekt	Rolle	Startdatum	Enddatum	Status
Diplomarbeit André Langer	Diplomand	01.05.2007	31.12.2007	ongoing

Von mir zu erledigende Aufgaben

Aufgabe	Projekt	in	Termin	Zuständigkeit	Abgeschlossen
Paper hochladen	Diplomarbeit André Langer	Paper schreiben	--.--.---- -	Alle	☐

Von Anderen zu erledigende Aufgaben

Momentan keine von Anderen zu erledigenden Aufgaben

Abbildung 6: Die Projektübersichtsseite für den individuellen Nutzer

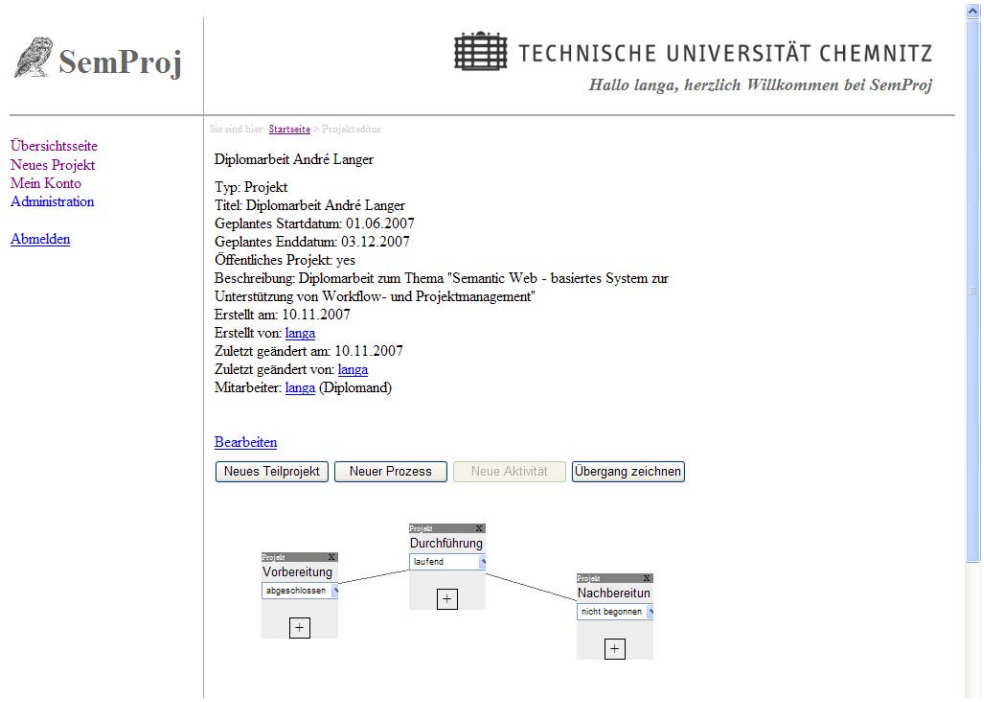


Abbildung 7: Der Projekteditor mit einem Informationsbereich und einem Modellierungsbereich. In dem Modellierungsbereich werden die einzelnen Objekte (Projekt, Prozess, Aktivität) unterschiedlich dargestellt. Jedes Objekt verhält sich wie in einem klassischen Fenstersystem und ist verschiebbar als auch durch ein Click auf das X-Symbol oben rechts schließbar. Des weiteren finden sich Schnellzugriffskontrollen zum Ändern des Objekttitels und des Objektstatus. Ein Symbol am unteren Rand des jeweiligen Objekts symbolisiert, dass es hierarchisch näher untergliedert ist.

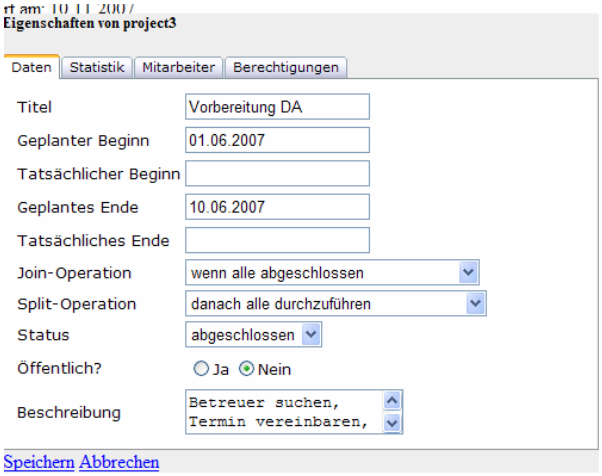


Abbildung 8: Eigenschaftenfenster zur Anpassung der Objekteigenschaften. Das Eigenschaftenfenster kann entweder durch Rechtsclick auf das betreffende Objekt oder durch Click auf die Bearbeiten-Schaltfläche im Informationsbereich geöffnet werden.

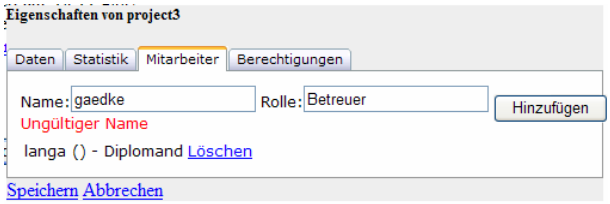


Abbildung 9: Eigenschaftenfenster zum Hinzufügen neuer Projektmitarbeiter

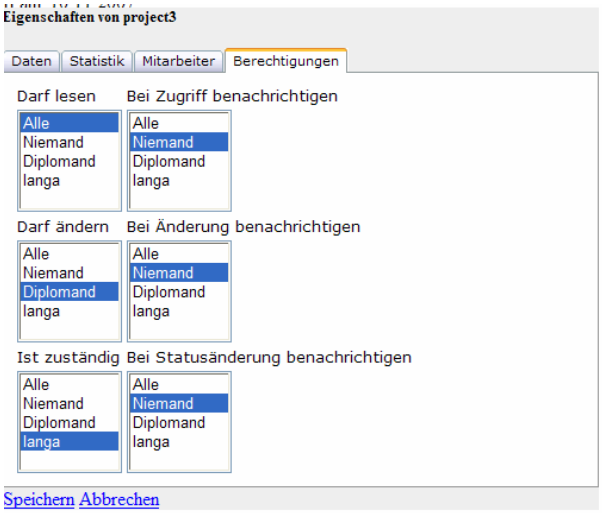
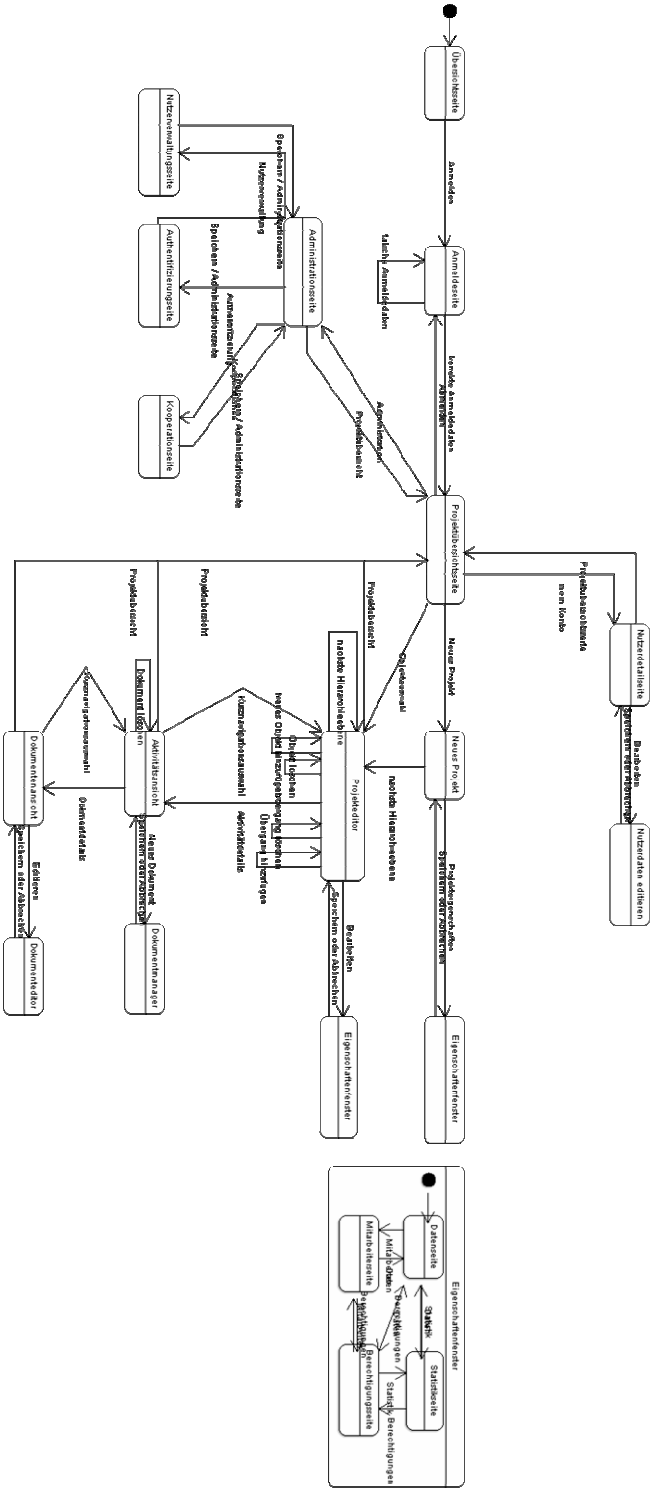


Abbildung 10: Eigenschaftenfenster zum Setzen der Zugriffsberechtigungen

Abbildung 11: Am Ende jeder Seite finden sich Informationen zur Datenquelle, aus dem die gerade betrachteten Informationen bezogen werden, sowie darunter ein Versionsverwaltungsinterface zur Herstellung vorhergehender Versionen. Rechts daneben finden sich zwei Schaltflächen, über welche die aus der Datenquelle extrahierten RDF/XML-Daten sowie der reine XHTML-Quelltext der Seite ohne RDFa-Annotationen angesehen werden kann.

- Zustandsdiagramm


Übersichtsseite
Anmeldedaten
Anmeldeseite
falsche Anmeldedaten
Administrationsseite
Nutzerverwaltungsseite
Authentifizierungsseite
Kooperationsseite
Projektübersichtsseite
Projektübersicht
Neues Projekt
Projektdetailseite
Projektdetailseite
Projektdetailseite

1.7. Datengrundlage

Zur Speicherung der Daten wird keine separate Datenbank benötigt. Ziel der zu entwickelnden Webapplikation ist es, dass diese auf herkömmlichen (X)HTML-Dokumenten operieren kann, aus denen alle benötigten Informationen extrahiert werden können. Obwohl die Speicherung von Daten auf Dateisystemebene aufwändiger ist und einige Nachteile in der alltäglichen Programmierung bietet (Unstrukturiertheit, fehlende Indexierung, Redundanzen, Inkonsistenzen und andere), soll mit dem zu entwickelnden Prototypen demonstriert werden, dass mit den Ideen des Semantic Webs Anwendungen erstellt werden können, welche Informationen aus beliebigen Datenquellen verarbeiten und kombinieren können. Im weltweiten Datennetz werden die meisten Daten als HTML-Dokumente ausgeliefert, ob nun physisch als Datei auf dem Servervorhanden oder serverseitig dynamisch generiert. Wesentlich ist nur, dass das jeweilige Dokument über eine entsprechende URL aufgerufen und gelesen (bzw. durch einen Internetbrowser dargestellt) werden kann. Die zu entwickelnde Webapplikation muss also in der Lage sein, sowohl semantisch annotierte HTML-Dokumente zu erstellen, als auch vorhandene Dateien unter einer bestimmten URL auslesen zu können, die als Datenspeicher verwendet werden.

Zur Datenauszeichnung kommen zwei unterschiedliche Konzepte zum Einsatz. Einerseits Microformats (vgl. http://www.microformats.org), andererseits RDFa (http://www.w3.org/TR/xhtml-rdfa-primer/) welches zukünftig zur semantischen Auszeichnung von Daten in XHTML 2.0 stark an Bedeutung gewinnen soll. Die Kombination beider Konzepte erfolgt aus zweierlei Gründen: Zum Einen sollen Vor- und Nachteile beider Formate im Rahmen einer Diplomarbeit näher untersucht werden, da sich beide für unterschiedliche Einsatzgebiete unterschiedlich gut eignen, sich andererseits mitunter gerade deswegen auch ergänzen und teilweise sogar ineinander überführt werden können. Andererseits zeichnet sich eine Entwicklung ab. Zum Anderen scheint das Ressource Description Framework (RDF) – Format als Basisformat unerlässlich zu werden, auf dem beliebige semantische Operationen durchgeführt werden können.

Microformats werden in diesem Zusammenhang dazu benutzt, Informationen in dem Workflow- und Projektmanagementsystem auszuzeichnen, welche gemeinsame charakteristische Eigenschaften besitzen und in andere Anwendungen sinnvoll exportiert werden können. Dies sind in erster Linie der bereits gut erforschte Bereich der Personeninformationen (hCard Microformat), welches im Kontext der an einem Projekt beteiligten Mitarbeiter verwendet werden kann als auch Kalenderinformationen (hCalendar Microformat), worüber Start- und Endtermine bereits vergangener oder noch durchzuführender Aktivitäten und Prozesse bequem ausgelesen und verarbeitet werden können. Daneben gab es Bestrebungen, weitere Microformats zur Auszeichnung von Projektinformationen wie hDOAP (http://dannyayers.com/xmlns/hdoap) oder hToDo (vgl. http://microformats.org/wiki/htodo) zu entwickeln, die jedoch noch alle nicht über einen Erstentwurf hinausgekommen sind und aus diesem Grund auch bisher nicht oder kaum von anderen Anwendungen unterstützt werden.

Wesentlich flexibler kann die Auszeichnung geschehen, indem eine auf den Anwendungsfall zugeschnittene Ontologie verwendet wird. Diese kann in RDFs ausgedrückt und als RDF/XML verarbeitet werden. Um diese semantischen Informationen in bestehende Dokumente einbetten zu können, wird der RDFa – Ansatz gewählt, wobei folgende Prädikate verwendet werden:

Abbildung 13: verwendete RDF-Prädikate

Physisch existieren hiervon die Objekte Projekt, Prozess, Aktivität, Dokument und Nutzer in jeweils eigenen Dateien unter einer eigenen Ressource URL. Die Konzepte Projektmitarbeiter, Projektrollen (einschließlich Berechtigungen) und Zustandsübergang existieren zusätzlich innerhalb dieser Dateien, da auf diese direkt referenziert wird, sie aber kein physisches Objekt der Realwelt nötigerweise darstellen müssen sondern nur in Verbindung mit anderen Objekten in beispielsweise einem Projekt auftreten.

2. Spezifikation der operationellen Anforderungen

2.1. Operationelle Anforderungen an die Daten

Für die Abschätzung des Speicherplatzbedarfs wird davon ausgegangen, dass alle Daten innerhalb eines eigenständigen HTML-Dokumentes als Kette von ASCII-Zeichen gespeichert werden, wobei zur Speicherung des Dokumentinhaltes pro Zeichen 1 Byte erforderlich ist.

Element	**Strukturbeschreibung**	**Bytes**
SemProjObjektDatei	= HTMLhead + HTMLbody	= 522 + prop*510 + obj*1530 + trans * 255 + 360 = 882 + prop*510 + obj*1530 + trans * 255
HTMLhead	= (xmldec) + (doctypedec) + htmltags + htmlheader	= 40 + 120 + 50 + 150 = 360
xmldec	“<?xml version=’1.0’ encoding=’utf-8’ ?>“	40
doctypedec	“<!DOCTYPE html …“	120
htmltags	“<html xmlns=’“ + string255 + “’></html>“	50
htmlheader	“<head profile=’http://www.w3.org/2003/g/data-view’><link rel=’transformation’ ref=’“+ string255 + “’/> </head>”	150
HTMLbody	= bodytags + inforarea + workarea	= 12 + 255 + prop*510 + 255 + obj*1530 + trans*255 = 522 + prop*510 + obj*1530 + trans * 255
Bodytags	“<body></body>”	12
Infoarea	= containertags + {propertydesc + propertyvalue} *InfoArea sei eine einfache Auflistung der Informationen über das dargestellte Projekt/Prozess/Aktivität/Dokument der Seite*	= 255 + prop*(255+255) = 255 + prop*510
containertags	string255 *Container-Element mit RDFa-spez. Namespacedeklarationen und about-Statement*	255
propertydesc	string255 *Zeichenkette für den Menschen zur Beschreibung der dargestellten Eigenschaft*	255
propertyvalue	string255 *Eigenschaftswert einschließlich property oder rel-Attribut*	255
Workarea	= containertags + {semprojobject} + {transitions} *workarea sei die grafische Darstellung einer Hierarchieebene eines Projektes/Prozesses/Aktivität*	= 255 + obj*1530 + trans*255
semprojobject	= objectcontainer + draghandle + objecttitle + objectstatus + changebutton + posinfo	=255+255+255 +255+255+255 = 1530
Objectcontainer*	String255 *bspw. ein div-Element zur grafischen Darstellung des Objekts*	255
Draghandle*	String255 *ein weiteres grafisches Element, um es beim Laden in	255

	die Anwendung zu verschieben*	
Objecttitle*	String255	255
Objectstatus*	String255 *grafische Darstellung des Status*	255
Changebutton*	String255 *ein weiteres grafisches Element zum Symbolisieren einer weitergehenden Hierarchie*	255
Posinfo	String255 *semantische Annotation, um die Positionsdaten des jeweiligen Objekts maschinenauswertbar zu hinterlegen*	255
Transitions*	String255 *Symbolisierung der Übergangskanten in dem jeweiligen Projektgraph*	255

Tabelle 3: Abschätzung des Speicherplatzbedarfs

Die mit * gekennzeichneten Elemente können weitere Daten wie z.B. Bilder einbinden, wodurch sich die Zeichenkette im Quellcode zwar nicht vergrößert, aber möglicherweise mehr Daten als Folge der Dokumentanforderung übertragen werden.

- **Gesamtbedarf pro Datei**

Geht man von einer einzelnen Projektdatei aus, so kann der Gesamtbedarf pro Datei abgeschätzt werden. In einer pessimistischen Schätzung enthalte diese beispielsweise 10 weitere dargestellte Projekt- oder Prozessobjekte, welche linear miteinander verkettet sind. Weiterhin seien in der Projektdatei 20 weitere Informationen enthalten, die das Projekt näher beschreiben. Die resultierende Projektdatei hat demzufolge eine Größe von

= 882 + 20*510 + 10*1530 + 9 * 255 = 882 + 10200 + 15300 + 2295 = 28677 Byte = 28kB

10 Objekte pro Hierarchieebene erscheint dabei schon viel, wobei sich eine tiefere hierarchische Anordnung dabei empfehlen würde. Die Gesamtanzahl an Teilprojekten, Prozessen und Aktivitäten ist schwer abzuschätzen und richtet sich stark nach Art und Umfang des durchzuführenden Projektes. Geht man von der pessimistischen Schätzung aus, dass ein Projekt aus 10 Teilprojekten (Phasen) besteht, die wiederum aus 10 Prozessen bestehen, die jeweils 10 Aktivitäten enthalten, so kommt man auf eine Gesamtdateianzahl für Projekte und Prozesse a 1 + 10 + 10*10 = 111 Dateien a 28kB. Die weiteren1000 Dateien mit Aktivitätsinformationen fallen kleiner aus, da diese zwar Dokumente enthalten können, diese jedoch als normale Verweisrelation gespeichert werden und es keine Übergangskanten zwischen diesen in einer grafischen Repräsentationsform zu speichern gibt. Eine Aktivitätsdatei mit zehn zugeordneten Dokumenten käme dabei dann auf eine Dateigröße von 882 + 30*510 = 15kB. Die referenzierten 10000 Dokumentdateien selbst wiederum enthalten zwar auch einen Dokumentkopf und Dokumentinformationen (882+20*510), sowie einen Dokumentinhalt (1000 Zeichen Fließtext), aber keine weiteren Verweise auf andere, durch das System verwaltete, aber noch nicht betrachtete Dateien.

Damit erhält man als pessimistische Abschätzung für den Gesamtspeicherplatzverbrauch eines Projektes inklusive aller referenzierten, untergeordneten Dateien

111 * 28kB + 1000 * 15kB + 10000 * 13kB = (3108 + 15000 + 130000) kB =144 MB.

Noch nicht betrachtet darin wurden die Nutzerdateien der am Projekt beteiligten Mitarbeiter. Geht man auch hier von 10 Partizipanten aus und der Tatsache, dass in der Regel die Mitarbeiter an einem Teilprojekt eine Teilmenge der Mitarbeiter an einem Gesamtprojekt darstellen, so fällt der Speicherplatzverbrauch der Nutzerdateien nicht näher ins Gewicht. Bedingt durch das dahinter liegende Versionierungssystem kann sich jedoch dieser Speicherplatzverbrauch weiter erhöhen, wenn bei jeder Dateimodifikation eine neue Dateiversion in einer separaten Datei gespeichert wird.

Die durchgeführte Rechnung stellt eine sehr pessimistische Schätzung dar, da der Aufwand, ein Projekt als Abfolge von 1000 Einzelaktivitäten in ein Computerprogramm einzutragen in dieser Form sicher nicht betrieben wird. Realistisch sind Einzelprojekte bestehend aus bis zu fünf Phasen (Teilprojekten) mit einer Abstraktion mehrerer darin enthaltener Prozesse in einer Aktivitätsbeschreibung und nur wenigen zugeordneten Dokumenten.

31 (Projekte und Prozesse) * 20 kB + 100 (Aktivitäten) * 14kB + 100 (Dokumente) * 13 kB = 3MB

- Übertragene Daten

Werden die Projektdateien durch das Projektmanagementsystem verarbeitet und in der Projekteditorumgebung dargestellt und zum Client übertragen, nimmt die übertragene Dateigröße aus folgenden Gründen zu

- Es werden weitere Seiteninhalte des Frontends übertragen, da die Informationen im Inhaltsbereich der Masterseite dargestellt werden
- Durch die Programmierung basierend auf ASP.NET 2.0 werden eine Reihe an Verwaltunginformationen im Quelltext dynamisch erzeugt und mitübertragen (beispielsweise Viewstate)
- Der Projekteditor basiert auf ASP.NET Ajax, wodurch eine Reihe an Stubs zum clientseitigen Aufruf der Serverfunktionen sowie weiterer Javascript-basierter Control-Elemente erzeugt werden

2.2. Operationelle Anforderungen an die Prozesse

In 2.1. ist ersichtlich geworden, dass an der Repräsentation eines Projektes eine Reihe an Dateien einbezogen werden muss, die verteilt abgespeichert werden und nicht einmal zwingend auf dem gleichen Server liegen müssen, sondern auf die über eine URL zugegriffen wird. Allein aus diesem Umstand heraus kann das zu entwickelnde System recht träge werden, was insbesondere bei Webapplikationen bei längeren Warte- und Verarbeitungszeiten problematisch werden kann.

Hinzu kommt, dass die Verarbeitung der einzelnen Projektdateien in einem zweistufigen Prozess erfolgt. Um RDF/XML-Tripel aus einem gültigen Dokument zu extrahieren, wird zunächst eine XSL Transformation (GRDDL) durchgeführt. Diese erhaltenen Tripel werden anschließend von einem RDF framework gruppiert und zusammengefasst, um darauf vereinfacht logische Operationen wie SPARQL-Anfragen durchführen zu können. Bei einem vorausgegangenen Test der RDF frameworks stellte sich diese Transformation bereits als recht zeitintensiv heraus, wodurch Verarbeitungszeiten von bis zu 2 Sekunden entstanden.

Zur Ermittlung verschiedener Projektinformationen ist im Gegensatz dazu eher damit zu rechnen, dass mehrere zum Projekt gehörige Dateien hintereinander verarbeitet werden müssen, um eine spezielle Information zu erhalten. Um dies in einem realistischen Rahmen umzusetzen, sind folgende Voraussetzungen zu erfüllen:

- Bereits gewonnene RDF/XML-Transformationsergebnisse werden in einem Cache zwischengespeichert, solang der Inhalt der Basisdatei nicht modifiziert wird, um unnötige Neutransformationen zu vermeiden
- Alle Inferenzoperationen sind so lokal wie möglich auszuführen, sodass nur direkt involvierte Dateien durchsucht werden müssen. Bei einigen Operationen (bspw. Auflistung aller Projekte, an denen die betreffende Person beteiligt ist), kann dies schwierig umzusetzen sein
- Ladevorgänge sind möglichst im Hintergrund auf dem Server auszuführen (AJAX). Anforderungen über einen Webservice mittels Ajax verbunden mit einer Fortschrittsanzeige können das subjektive Warteempfinden verringern, als wenn bei jeder Anforderung die komplette Seite neu geladen werden muss.

Antwortzeiten von bis zu 5 Sekunden müssen aus diesem Grund toleriert werden. Weitere Szenarien, für die extremere Anforderungen an die Prozesse gestellt werden müssen, gibt es nicht.

3. Spezifikation wichtiger Qualitätsanforderungen

3.1. Benutzbarkeit

Vom Nutzer des Systems werden grundlegende Kenntnisse im Umgang mit Computern und Programmen vorausgesetzt. Die Oberfläche und das Nutzerinterface ist so konzipiert, dass es überschaubar und weitestgehend selbsterklärend ist. Ein Schwerpunkt ist auf eine einfache, intuitive Bedienung zu legen, die sich an einem desktoptypischen Verhalten orientiert, sodass jeder Nutzer das System ohne langfristige Schulungsmaßnahmen bedienen kann.

Das System selbst hat alle Pflichtkriterien aus Kapitel 1.4. zu erfüllen und soll in diesem Rahmen auch bedienbar und benutzbar sein. Da es webbasiert bisher kein vergleichbares System gibt, ist die durchzuführende Entwicklung als Prototyp zu sehen, auf den bei erfolgreicher Erprobung ein zuverlässigeres System mit weniger Einschränkungen entwickelt werden kann.

Die Installation des Systems erfordert zusätzliche Kenntnisse zur Veröffentlichung und Bereitstellung einer Webanwendung auf einem Internet Information Server.

3.2. Zuverlässigkeit

Bei der Bedienung der Anwendung ist dafür zu sorgen, dass unvollständige oder fehlerhafte Eingaben überprüft werden und der Anwender darauf hingewiesen wird. Dazu sollte die Gestaltung der Benutzeroberfläche unter ergonomischen Gesichtspunkten diese Kontrolle unterstützen.
Webapplikationen unterliegen generell gewissen Einschränkungen, dass eine einhundertprozentige Verfügbarkeit nicht garantiert werden kann. Es sollte jedoch sichergestellt werden, dass das System bei korrekter Bedienung nicht in einen nicht-definierten Systemzustand geraten kann und im Problemfall dieser erkannt und einfach verlassen werden kann. Für die Konsistenz der Daten wird durch ein im Hintergrund liegendes Versionsmanagementsystem gesorgt, welches alle Änderungen regelmäßig sichert.

3.3. Integrität

Aufgrund der Aufgabenbeschreibung, ein prototypisches System zu entwickeln, bei dem die Realisierbarkeit eines semantik-basierten Ansatzes im Workflow- und Projektmanagementbereich überprüft wird, sind keine besonderen Anforderungen an die Integrität der Daten zu stellen. Alle Dateien sind im Dateisystem des Servers frei zugänglich, wobei der Zugriff durch zusätzliche Maßnahmen jedoch eingeschränkt werden kann. Eine systemunabhängige Modifikation und Manipulation ist prinzipiell möglich, wird aber nicht empfohlen.

Es existiert lediglich eine Integritätsbedingung, dass pro Verzeichnis jede Datei eine eindeutige Identifikationsnummer besitzt, beispielsweise /projekt/projekt-29.htm

3.4. Flexibilität

Prinzipiell ist es durch den objektorientierten Ansatz des Systems möglich, das Programm auch in Zukunft weiterzuentwickeln. Da die Anforderungen an das System wie bei jedem größeren Projekt zu Beginn jedoch unklar waren und anzunehmen ist, dass diese sich während der Entwicklung noch verändern werden, wird für einen industriellen Einsatz empfohlen, die gewonnenen Erkenntnisse aus dem zu entwickelnden System in einer Neuimplementierung weiter zu verwenden. Gegenwärtig sind keine zukünftigen Weiterverwendungsansätze des Systems bekannt, sodass bezüglich der Flexibilität keine zusätzlichen Anforderungen gestellt werden.

3.5. Portabilität

Eine Portierung des Systems auf Serverumgebungen, welche sich von der Entwicklungsumgebung unterscheiden, ist nicht vorgesehen. Da es sich um eine webbasierte Anwendung handelt, ist eine plattformübergreifende Nutzung des Systems allgemein sichergestellt.

4. Basismaschine und Entwicklungsumgebung

4.1. Hardware- und Softwareanforderungen

Folgende Entwicklungsumgebung der Webanwendung wird serverseitig vorausgesetzt, die sich an den notwendigen Minimalanforderungen einer ASP.NET 2.0 – Applikation orientiert:

- PC/Server mit einem Prozessor neuerer Generation (Pentium/Athlon, 2GHz empfohlen) und einem Arbeitsspeicher von mind. 512MB
- Festplatte mit ausreichend Speicherplatz von mind. 1GB
- Windows Server 2k mit installierter ASP.NET 2.0 Laufzeitumgebung
- Internet Information Server 5.0 oder neuer
- ASP.NET Ajax Extensions im GAC, ASP.NET Ajax Control Toolkit vorhanden
- Internetanbindung mit mind. 2MBit

Clientseitig wird ein handelsüblicher PC mit folgender Konfiguration empfohlen:

- PC mit einem Prozessor neuerer Generation (Pentium/Athlon, mind. 1.5GHz) und einem Arbeitsspeicher von mind. 256MB
- 17’ Farbmonitor mit einer Mindestauflösung von 800x600px oder höher
- Tastatur, Maus
- Internetbrowser neuerer Generation (Internet Explorer 6.0 oder höher, Mozilla und alle anderen auf der Gecko-Engine basierenden Browser ab Version 1.0, Opera ab Version 7.6. u.a)
- Javascript und Cookie-Annahme aktiviert
- Gängige Internetanbindung mit mind. 128kbit/s

Damit lassen sich alle Anforderungen an die zu entwickelnde Software komplett erfüllen. Durch den Einsatz einer leistungsfähigeren CPU, mehr Arbeitsspeicher und eine schnellere Internetanbindung kann das Antwortzeitverhalten der Webapplikation verbessert werden.
Ein größerer Monitor mit höherer Auflösung ist erstrebenswert, um die Arbeitsbedingungen des Nutzers zu verbessern, da das System auch über einen längeren Zeitraum genutzt werden kann.

Zusätzliche Software wird nicht benötigt.

4.2. Entwicklungsumgebung

Zur Umsetzung der Webanwendung wird das Microsoft Visual Studio 2005 verwendet. Zur Erstellung der UML-Diagramme in der Analyse- und Designphase wurde ArgoUML verwendet.

5. Systemarchitektur

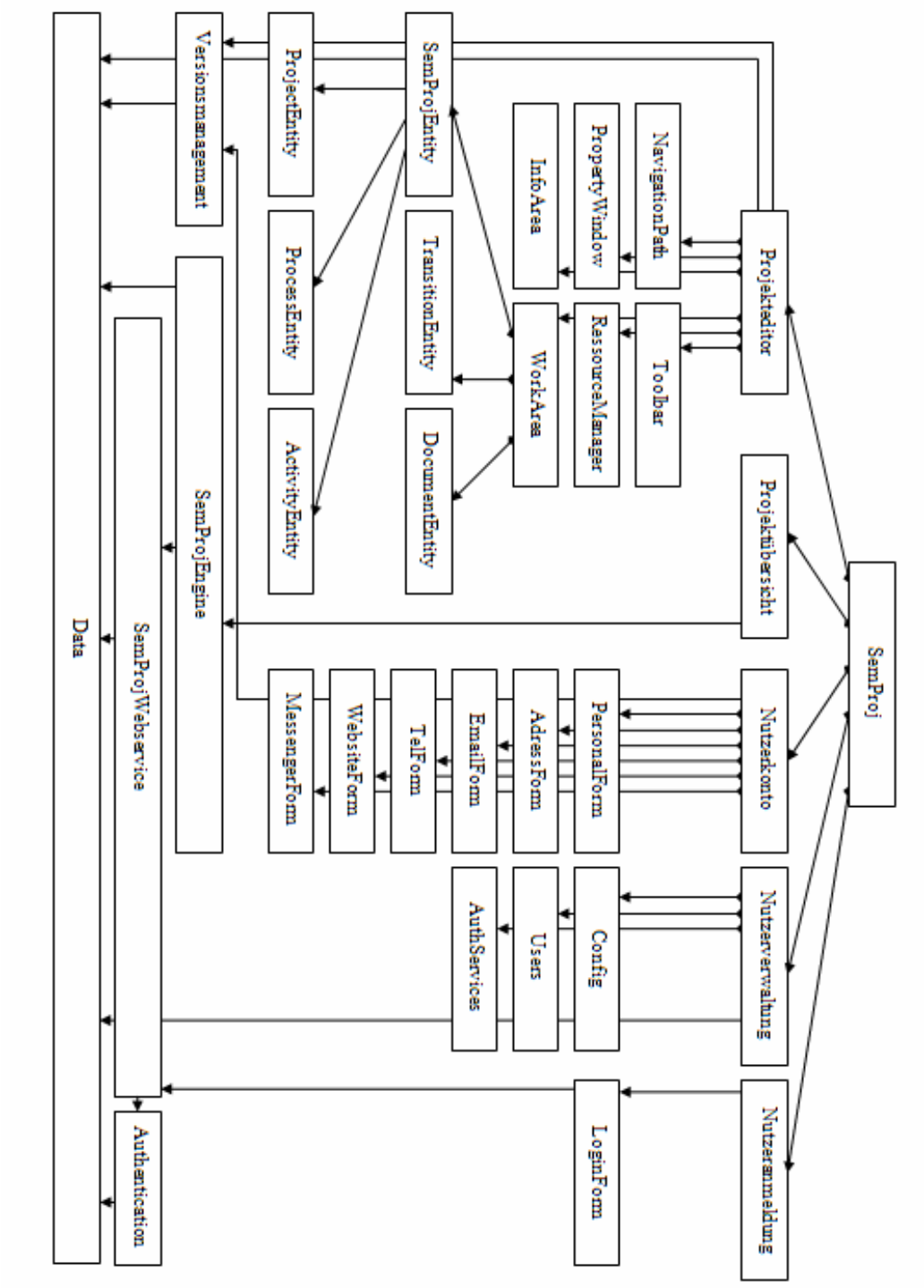


Abbildung 14: schematische Systemarchitektur

6. Klassenspezifikation

Abbildung 15: Klassendiagramm